młody TECHNIK

# InforMik

MAGAZYN KOMPUTEROWY „MŁODEGO TECHNIKA"

III
1988



## Nasz komentarz

# STRASZNY DWÓR ALBO STUDIO KOMPUTEROWE

Felieton o spolszczonym oprogramowaniu wywołał silny rezonans Czytelników. Opinie były zróżnicowane i chociaż zdecydowanie przeważał realizm, czasem zaprawiony goryczą, to trafił się też Czytelnik, który, i owszem, podjąłby ze mną dysputę, wszakże tylko pod warunkiem, że będzie odbywać się ona przez kratki (aresztu, ma się rozumieć). Dzisiaj tematu ciąg dalszy.

Aby nie było nieporozumień: uważam, że ustawowa ochrona praw autorskich jest niezbędna. Jako praktyk jestem jednak pesymistą co do jej skuteczności, a zwłaszcza efektów ekonomicznych. Osobiście nie spodziewam się w wyniku wprowadzenia jakiejkolwiek ustawy zalewu rodzimym oprogramowaniem, zwłaszcza zaś oprogramowaniem narzędziowym.

Z szacownych kręgów notabli oficjalnych instytucji informatycznych padają bardziej optymistyczne opinie. Chętnie uwierzyłbym profesorom bez zastrzeżeń, jednak przeszkadza mi w tym zakalcowaty guz umownie zwany bułką, który próbuję przeżuwać pisząc ten tekst. Przed kilku laty pewien profesor, też ekspert wielkiej miary, roztaczał przed nami miraż chrupiącego pieczywa i uzasadniał nim konieczność innych pociągnięć prawno-administracyjnych.

Obawiam się, że sama słuszność nie stanowi żadnej gwarancji praktycznych sukcesów. Potrzebny jest kontakt z tzw. życiem i pewna wobec tego życia pokora. Nawet jeżeli wydamy przepisy o ochronie obcych praw autorskich, to i tak większość użytkowników w Polsce będzie posługiwać się (zdobytym raczej mniej niż bardziej legalnie) oprogramowaniem zachodnim. Chociażby dlatego, że jest ono bardzo bogate i reprezentuje wysoki poziom, który trudno będzie w kraju osiągnąć pomimo najlepszych nawet ustaw. Jeśli tak, to w dalszym ciągu będzie potrzebna polskojęzyczna dokumentacja oraz pakiety adaptujące zagraniczne programy do specyfiki języka polskiego. Muszą to być opracowania na odpowiednim poziomie, dostępne wraz z towarzyszącym im doradztwem technicznym i innymi niezbędnymi specjalistycznymi usługami.

Można oczywiście (w imię słuszności) problemu nie dostrzegać, ale czy to będzie oznaczało jego likwidację? Łatwo stwierdzić, że natura nie znosi próżni. Trudniej dostrzec, czym natura próżnię wypełnia. Przykład? Służę uprzejmie: nasz rodzimy rynek literatury informatycznej i oprogramowania.

O tym, że do komputera potrzebne jest oprogramowanie i dokumentacja, wie już każdy. Tworzeniem nowych programów i adaptacją zagranicznych zajmuje się coraz więcej zarówno firm, jak i osób prywatnych. Nie brak też osób i instytucji, tłumaczących mniej lub bardziej udolnie zagraniczne opracowania literaturowe lub nawet piszących własne, o cechach oryginalności. Problemem jest natomiast brak profesjonalnej organizacji handlowej o szerokim zasięgu, zapewniającej z jednej strony szeroką informację i łatwość zakupu po rozsądnych cenach dla zainteresowanych, jak i opłacalność dla twórców. Dzieje się to w warunkach dużego i stale rosnącego popytu. Sytuacja wprost wymarzona dla cwaniaków i wydrwigroszy.

Wystarczy dysponować lokalem i niewielkim kapitałem, a także gromadą małolatów, którzy pokątnie i za psi pieniądz będą nagrywać kasety z grami dla komputerów domowych. Potem jeszcze stosowne ogłoszenie, plakat: Studio komputerowe oferuje... i interes może się rozkręcać. Przypuśćmy, że nasze studio mieści się w bliżej nie określonym centrum przemysłowym na północy Polski, a jego bossem jest kędzierzawy grubas, zwany Jojo.

Jojo rozpoczyna karierę od prywatnej klienteli dysponującej komputerami domowymi, zapewniającej nieewidencjonowany dopływ żywej gotówki. Jojo widzi jednak, że prawdziwy interes można zrobić na oprogramowaniu i literaturze profesjonalnej. Nadaje więc swemu półlegalnemu studiu status przedsiębiorstwa, uzyskuje stosowne zezwolenie i pieczątkę, po czym wypływa na szersze wody. Brak szerszego pojęcia o informatyce wcale mu w tym nie przeszkadza, podobnie jak fakt, że jego biuro mieści się w sypialni.

Podstawowym atutem Jojo jest całkowity brak skrupułów, wstydu, a także godności, jeśli tylko miałoby mu to przynieść wymierną korzyść. Kopiuje wszystko, co wpadnie mu w rękę, nie bacząc na żadne prawa autorskie, nawet wtedy, gdy ma do czynienia z autorem krajowym, ba, nawet kolegą. W ofercie Joja pełno jest pozycji, które już z daleka wyglądają na powielane nielegalnie. Jojo często nie zadaje sobie nawet trudu, aby zmienić stronę tytułową z nazwiskiem autora bądź nazwą legalnego pośrednika, tylko kopiuje jak leci. Bardzo chętnie wchodzi też w układy z autorami różnych opracowań, obiecując im złote góry. Da mu to później pretekst do trąbienia, iż jest autoryzowanym dystrybutorem, itd. Natomiast umówione honorarium autor najczęściej ogląda tylko we śnie.

Trzeba lojalnie przyznać, że Jojo jest energiczny i ruchliwy. W połączeniu z bezczelnością i tupetem, a także rozmaitymi układzikami, daje mu to niezłą siłę przebicia. Jojo bardzo dba o pozory, tzw. bajery, na które można wziąć klienta jak muchę na lep. Obowiązkową dekoracją biurka Joja jest, obok kwiatka i popielniczki, atrakcyjna młoda dama, której główną kwalifikacją informatyczną jest umiejętność uśmiechania się. Z resztą personelu jest już gorzej: stanowi go kilku młodocianych subiektów do podawania kaset i dyskietek, paradujących w (obowiązkowo) kraciastych gaciach.

Jojo troszczy się o reklamę. Przed każdą większą imprezą stara się zwrócić na siebie uwagę prasowymi artykułami. Czasem sygnuje je osobiście (materiał pisze mu rzecz jasna kto inny), innym razem zleca napisanie „obiektywnej" relacji o firmie. Zwitek banknotów loco kieszeń sprawia, że najemny pismak dwoi się i troi, aby nadać mitowi studia pewne cechy prawdopodobieństwa. Swoistym majstersztykiem są prasowe wywiady z Jojem. Znajomi Joja wiedzą, że ma on pewne trudności z poprawną polszczyzną, abstrahując od faktu, że przedmiot swej działalności zna głównie przez pryzmat cennika. Cóż to dopiero za ubaw przeczytać w gazecie pełne namaszczenia, świetliste myśli Joja! Jest pewne, że Jojo powinien uważnie czytać wywiady z samym sobą, gdyż mógłby się z nich sporo dowiedzieć.

Konkluzja: jeżeli reklama i pozory wskazują, że macie do czynienia z godnym szacunku komputerowym ekspertem i solidnym partnerem, a przy pierwszym spotkaniu jawi wam się bełkotliwy gamoń, nie miejcie złudzeń — prawdziwe jest to drugie oblicze. Wątpliwości może rozwiać kilka pytań o konkrety. Jeśli podejrzenie się potwierdzi, bierzcie nogi za pas i zmykajcie, gdzie pieprz rośnie — to właśnie w kantorku Joja, nie zaś w atmosferze uroczystych konferencji tkwi prawda o przestrzeganiu w Polsce praw autorskich. Zjawiska Joja nie zniszczą najlepsze przepisy. Zepchnąć go na margines może jedynie rzetelna konkurencja, operująca atutem jakości oraz wzrost świadomości i wrażliwości na kulturę, a także fachowość obsługi u potencjalnych klientów.

**Roland Wacławek**

*P.S. Gruby Jojo i jego kantor są obiektami całkowicie fikcyjnymi, wymyślonymi dla potrzeb niniejszego artykułu.*

**Tę barwną grafikę wydrukowano na nowoczesnej drukarce firmy „Citizen" MSP-55 podczas wystawy wyrobów tej firmy w Warszawie**



młody TECHNIK

InforMik

MAGAZYN KOMPUTEROWY
MŁODEGO TECHNIKA
NR III (6) ROCZNIK II

SPIS TREŚCI



**Numer ilustrował** Roman Gaik
**Zdjęcia w numerze:** „Byte", W.P. Jabłoński, T. Rzepecki, G. Zalot

# ŻÓŁTE, MIGAJĄCE...

*Moda na komputery przeminęła, a zaczął się okres pracy z komputerem. I tu nagle okazało się, że machina biurokratyczna rozpoczęła kontrofensywę przeciw „szybkiemu wariatowi" w miejscach pracy. Oto kilka przykładów:*

*W pewnych zakładach przemysłowych komputer wykazywał tyle spóźnień, że płace robotników (również wyliczane przez komputer) i premie spadły tak drastycznie, iż połowa załogi chciała się zwolnić, więc ze względów „społeczno-politycznych" komputer skasowano...*

*W pewnej firmie kilku specjalistów zaczęło wykonywać projekty za pomocą Commodore'a i dostarczać swoje wyliczenia w postaci wydruków (prosty program umożliwiał wykonanie pracy w ciągu... 8% dawnego czasu). Nie podobało się to jednak panu dyrektorowi firmy, który najpierw przeliczał „na piechotę" wyniki komputerowe, a gdy błędu nie znalazł, zabronił używania komputera ze względów etyczno-moralnych, bo... będą mieli za dużo wolnego czasu w pracy, będą bałaganić, a inni, co to nie mają komputera, będą na tym pokrzywdzeni. Na sugestię skomputeryzowania całej firmy (Commodore był prywatny, co też uznano za przestępstwo!) padła odpowiedź, że firma ma — i owszem — pieniądze, ale dla fanaberii panów inżynierów nie będzie ich... wyrzucać w błoto...*

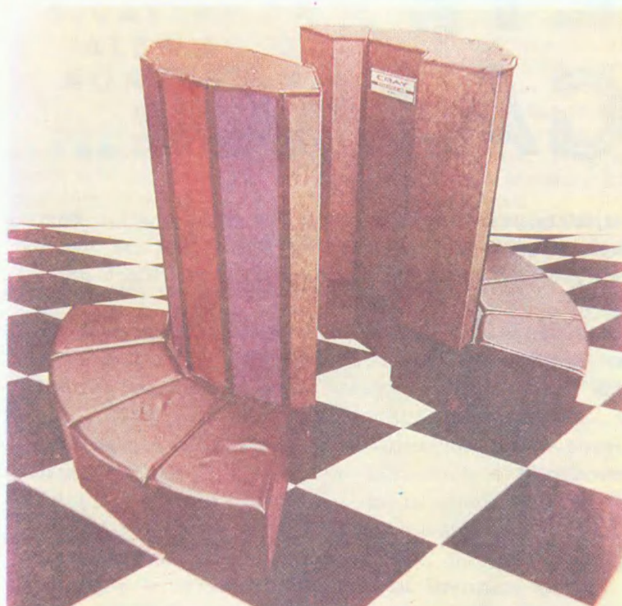
*Pewien znajomy odwiedzał bardzo wysoko postawioną osobistość. W sekretariacie dowiedział się, że osobistość właśnie ciężko pracuje i musi, chcąc nie chcąc, zaczekać. W trakcie oczekiwania znajomy ów usłyszał z gabinetu odgłosy dobrze mu znanej gry na komputer IBM. Motyw muzyczny był tak wyraźny, że o pomyłce nie ma mowy...*

*(Znajomi znajomego zafundowali sobie komputerowe wydanie „Kamasutry"! Zawsze mówiłem, że grafika komputerowa może pomóc „niepełnosprawnym"!!!)*

*Jak więc naprawdę jest z komputeryzacją w Polsce? Podobno zapalono dla niej zielone światło — sprzęt komputerowy można przywozić bez cła, mnożą się firmy handlujące owym sprzętem i oprogramowaniem, wiele firm „nabywa drogą kupna" komputery, przedstawiciele świata przestępczego wiedząc co dobre okradają z komputerów Politechnikę Warszawską... Uff, chyba jest dobrze?*

*Nie, nie jest dobrze! Nie dorośliśmy jeszcze jako społeczeństwo do pełnej komputeryzacji. Dublowanie ksiąg finansowych na potrzeby organów kontrolnych zniechęca np. instytucje do prowadzenia bilansów komputerowych w sprawach finansowych. Co to za oszczędność, jeśli przychodzący kontroler nie potrafi sprawdzić bilansu sporządzonego przez komputer i musi mieć jeszcze zwykłą dokumentację, aby wynik mógł być skontrolowany! Nie da się uniknąć* ***zwiększenia*** *personelu, bo oprócz operatora komputera trzeba jeszcze zatrudnić klasyczny zestaw księgowych. Stosowanie komputera do kontroli technicznej jest przy naszej jakości produkcji niebezpieczne dla zakładu, jeśli bowiem coś jest nie tak, to komputer nie „przymknie oka". Nie palmy więc w prasie, radio i telewizji „zielonego światła" dla komputerów. Skorzystajmy z doświadczeń Stołecznego Urzędu Spraw Wewnętrznych, który wprowadził tytułem eksperymentu żółte, migające światło w sygnalizatorach świetlnych w nocy. Można więc nocą jechać płynnie przez całe prawie miasto, trzeba jednak pamiętać, że żółte światło wprawdzie pozwala jechać, ale i nawołuje do ostrożności. Zapalmy więc komputerom światło — może nie zielone, ale żółte, migające...*

*JERZY KLAWIŃSKI*



— rozwiązywanie zagadnień statyki i dynamiki budowli. Znalezienie sił w obciążonej ramie za pomocą tzw. metody przemieszczeń wymaga rozwiązania układu tylu równań, ile jest w niej węzłów. W konstrukcji wieżowca mogą być setki węzłów.

Budową i oprogramowaniem superkomputerów zajmują się dziesiątki tysięcy ludzi na całym świecie. W ciągu dwudziestu lat badań znaleziono trzy podstawowe schematy architektury superkomputerów: potokowy, wektorowy i asocjacyjny.

Przykładem maszyny asocjacyjnej jest amerykański system STARAN. Składa się on z tzw. matryc skojarzeniowych w postaci pól o wymiarach 256×256 bitów. W systemie może być do 32 matryc. Każdej matrycy przydzielono 256 jednostek procesorowych — po jednej na słowo. Do przetworzenia mogą być pobierane całe słowa lub ich wycinki o długości określonej przez programistę. Słowa mogą być umieszczone zarówno wzdłuż kolumn, jak też wierszy macierzy.

Unikalny jest sposób adresowania słów w macierzy. Przed wykonaniem operacji pobrane słowo jest przekazywane do specjalnego rejestru. Następnie zawartość rejestru jest porównywana z zawartością pól matrycy. Wynik operacji jest zapisywany wszędzie tam, gdzie zawartości okazały się identyczne. Jednocześnie są sprawdzane wszystkie słowa macierzy, więc czas znajdowania odpowiednich miejsc nie jest dłuższy niż przy zwyczajnej adresacji według miejsca w pamięci.

ADAM NOWICKI

# ANATOMIA SUPERKOMPUTERA

Zapewne większość naszych Czytelników fascynują przede wszystkim nowości z dziedziny mikrokomputerów. Dziś jednak zajmiemy się światem największych maszyn liczących — superkomputerów. Są to urządzenia, których stosowanie jest siłą napędową wielu dziedzin nauki i techniki.

Wymienię kilka przykładowych zastosowań superkomputerów:
- zarządzanie dużymi bankami informacji;
- prognozowanie pogody. Opracowanie prognozy średnioterminowej wymaga rozwiązania układu kilkuset równań różniczkowych, w których zmiennymi są na przykład temperatury czy prędkości wiatru;
- badania naukowe w fizyce, chemii, astronomii, biologii (symulacja zachowania łańcuchów genetycznych);
- analiza wyników poszukiwań geologicznych;
- symulacje zjawisk aerodynamicznych przy projektowaniu samolotów;
- symulacje badań projektów karoserii samochodowych w tunelach aerodynamicznych;
- projekty rozmieszczeń elementów w układach elektronicznych. W tej chwili wszystkie nowe układy cyfrowe są projektowane za pomocą komputerów — symulacja umożliwia przetestowanie działania przed budową prototypu;
- modelowanie systemów bezpieczeństwa reaktorów atomowych;
- wytwarzanie i obróbka obrazów graficznych (artykuł na ten temat zamieszczony był w „MT" 3/86);

Matryce mogą być połączone szeregowo i pracować w systemie potokowym, o którym napiszę za chwilę. Pamięć operacyjna komputera STARAN ma pojemność 8 megabajtów.

Architektura asocjacyjna może być stosowana tam, gdzie mamy do czynienia z dużą liczbą napływających równocześnie, szybko się zmieniających danych. Dobrym przykładem mogą tu być komputery sterujące odczytem danych czujników, wyznaczających trajektorie lotu pocisków balistycznych. Najczęściej są to maszyny specjalizowane, zbudowane w konfiguracji dostosowanej do wykonywania stale tego samego zadania. Mogą być szybsze od komputerów ogólnego przeznaczenia — specjalizacja ma swoje zalety.

Dalej jednak zajmę się maszynami uniwersalnymi, stosowanymi znacznie częściej.

Dużą wadą architektury asocjacyjnej jest trudność pisania programów dla komputera. Proponuję przerwać na chwilę lekturę i wyobrazić sobie konieczność przewidzenia, co uzyskamy w każdym z 256 procesorów przy różnych długościach pobieranych wycinków słowa... Właśnie dlatego komputery potokowe i wektorowe są budowane znacznie częściej.

Pracę komputera o architekturze potokowej wyjaśnię na przykładzie. Załóżmy, że należy wykonać fragment programu w Fortranie:

```
DO 10 I = 1,50
```

```
A(I) = B(I)*3+5
10 CONTINUE
```

Komputer wyposażony w jeden procesor musi 50 razy wykonywać mnożenie, 50 razy dodawanie i jeszcze poświęcić trochę czasu na 50-krotne zamknięcie pętli. Jeśli, tak jak procesory systemu Cray, na dodawanie zmiennoprzecinkowe musi poświęcić 6 cykli zegara, a na mnożenie 7 cykli, całą pętlę wykona w ciągu 650 cykli zegarowych. Czas ten wydłużą jeszcze bardziej operacje związane z zamykaniem pętli.

Można jednak zastosować dwa procesory połączone szeregowo. Gdy jeden procesor będzie wykonywał mnożenie dla elementu I tablicy, drugi w tym samym czasie wykona dodawanie dla elementu I-1. Potrzeba tylko $7\times50 = 350$ cykli na 50 mnożeń i dodatkowo 6 cykli na dodawanie dla 50. elementu (w tym czasie procesor mnożący nie ma już nic do roboty). Zastosowanie osobnej jednostki zliczającej wykonane operacje pozwala nie tracić czasu na zamykanie pętli.

Przykład jest prosty; dla bardziej złożonych obliczeń można uzyskać jeszcze większą oszczędność czasu. Odbywa się to kosztem sprzętu — potrzebujemy wielu procesorów, rozbudowanych układów sterujących przesyłem danych i wielu szybko dostępnych rejestrów.

Procesory możemy również połączyć równolegle. W ten sposób otrzymujemy procesor wektorowy — wykonujący równocześnie obliczenia dla wszystkich elementów jednowymiarowej tablicy. W praktyce rozmiary macierzy są większe od liczby procesorów, ale można je podzielić na kilka mniejszych wektorów. Nasz przykładowy programik zostanie wykonany w ciągu $7+6 = 13$ cykli zegara — kosztem jeszcze większych wymagań sprzętowych niż przy architekturze potokowej.

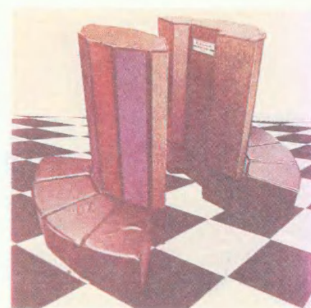
Pierwszym prawdziwym superkomputerem był **Cray-1**, zainstalowany w 1976 roku w laboratoriach nuklearnych w Los Alamos. Sercem systemu było 13 jednostek operacyjnych podzielonych na cztery grupy. Jednostki te mogły działać równocześnie, zajmując się różnymi zadaniami, lub w trybie przetwarzania potokowego. Poniżej zestawiłem grupy jednostek wraz z liczbą przydzielonych im rejestrów i wykonywanymi zadaniami:

— Jednostki adresowe. Operowały na 24-bitowych adresach dokonując ich modyfikacji oraz indeksowania. Rejestry mieściły 64 adresy.

— Jednostki skalarne, operujące na 64-bitowych liczbach całkowitych. Wykonywały operacje arytmetyczne i logiczne na wartościach z ośmiu rejestrów skalarnych. Jednostkom został podporządkowany bufor mieszczący 64 liczby.

— Jednostki zmiennoprzecinkowe wykonujące operacje na liczbach rzeczywistych. 64-bitowe słowo podzielono na 48-bitową mantysę (dokładność 14 cyfr znaczących) i 16-bitowy wykładnik umożliwiający zapis liczb z zakresu od $10^{-2500}$ do $10^{2500}$. Jednostki operują na argumentach pobranych z rejestrów skalarnych lub wektorowych.

— Jednostki wektorowe o długości słowa 64 bity wykonują działania arytmetyczne na liczbach całkowitych lub zmiennoprzecinkowych oraz operacje logiczne. Dysponują ośmioma akumulatorami mieszczącymi po 64 słowa. 64-bitowy rejestr maski określa elementy wektora, na których ma być wykonana kolejna operacja.



Jednostki w gruncie rzeczy odpowiadają architekturze potokowej — działania na elementach wektora nie są wykonywane równocześnie (oszczędności czasu uzyskuje się dzięki szybkiemu dostępowi do pojemnych rejestrów i organizacji pętli). Specjalny 7-bitowy rejestr definiuje długość aktualnie przetwarzanego wektora.

Pamięć operacyjna **Craya-1** mieściła 1048576 słów 64-bitowych uzupełnionych ośmioma bitami korekcji błędu. Została podzielona na 16 bloków, w których adresacja przebiega modulo 16. Jeśli więc pierwsze słowo pierwszego bloku nosi numer 0; to numer 1 będzie nosiło pierwsze słowo drugiego bloku a drugie pierwszego — 16. Dzięki temu w ciągu cyklu dostępu do pamięci trwającego 50 nanosekund można pobrać 16 kolejnych słów z instrukcjami i umieścić je w czterech buforach instrukcji mieszczących po 64 słowa. Można obliczyć, że szybkość przesyłu instrukcji wynosi 320 Msłów/s. Dane są pobierane z szybkością 80 Msłów/s.

Pamięć zewnętrzna systemu w Los Alamos składała się z trzech zespołów. Pierwszy stanowiły jednostki dysków magnetycznych o pojemności 4.75 gigabajta i średnim czasie dostępu 35 ms. Drugą — zestaw pamięci taśmowych o pojemności 63 gigabajty i czasie dostępu 20 sekund. Ostatni był biblioteką składającą się z kilku tysięcy taśm mogących łącznie pomieścić 250 000 gigabajtów informacji. W tym przypadku czas dostępu zależał od bystrości panienki z obsługi.

Komunikacja **Craya-1** z otoczeniem odbywała się poprzez 12 kanałów wejścia i 12 wyjścia, z których każdy mógł przesyłać informacje w tempie 10 Msłów/s. Wąskim gardłem operacji wejścia/wyjścia jest dostęp do pamięci — równocześnie może pracować tylko osiem kanałów. W systemie w Los Alamos zarządzaniem kanałami we/wy zajmował się mini-komputer ECLOPSE. W nowszych modelach superkomputerów Cray stosowany jest popularny na Zachodzie mini-komputer VAX 11/780.

Jednostkę centralną komputera zbudowano z układów scalonych mieszczących po 16 bramek logicznych NAND opóźniających przebieg sygnału o 1 nanosekundę. Zastosowano tranzystory bipolarne w technologii ECL (ang. Emitter Coupled Logic — emitery tranzystorów są połączone ze sobą). Pamięć operacyjną zbudowano z układów statycznych o organizacji $1K\times1$ bit. Dzięki sporej gęstości upakowania elementów i cylindrycznemu ułożeniu modułów udało się skrócić największą długość połączenia do 4 stóp (ok. 122 cm). Pozwoliło to na zastosowanie zegara o częstotliwości 84 MHz — długość cyklu zegarowego wynosi 12 nanosekund.

Ciepło wydzielane przez pracujące układy scalone jest w przybliżeniu proporcjonalne do kwadratu częstotliwości zegara. Dlatego należało zastosować system chłodzenia ciekłym freonem, wymagający dwóch kompre-



sorów o masie 25 ton każdy. Część układów chłodzących umieszczono wokół cylindra z procesorami tworząc wygodne siedzenia — dlatego komputery Cray zyskały sobie miano „najdroższych kanap świata". Układ zasilania o mocy 150 kW zapewnia dużą stabilność napięcia i bezawaryjną pracę.

Wydajność superkomputera zależy od zadania, które należy wykonać. Najszybciej realizowane są operacje na dużych macierzach. **Cray-1** w ciągu sekundy może wykonać od kilku do ponad stu milionów arytmetycznych operacji zmiennoprzecinkowych. Aby ujednolicić jednostki miary w laboratoriach w Livermore opracowano zestaw czternastu testów napisanych w języku Fortran. Od tej pory moc obliczeniową superkomputerów określa się jako średnią wydajność zmierzoną we wszystkich testach. Podaję dwa najkrótsze z nich — bez linii deklarujących zmienne:

```
   DO 1 K = 1,400
1 X(K) = Q+Y(K)*(R Z(K+10) + T*Z(K+
+11))
   DO 3 K = 1,1000
3 Q = Q+Z(K)*X(K)
```

Z przeprowadzonych testów wynikło, że moc obliczeniowa **Craya-1** wynosi 26 MFLOPS. Rozwiązanie nieliniowych równań opływu powietrza wokół skrzydła samolotu, przy użyciu IBM 360 trwające 18 godzin, superkomputerowi zajęło niecałą minutę. Tak dobry wynik osiągnięto częściowo dzięki dużej pojemności pamięci operacyjnej — komputer nie musiał tracić czasu na ściąganie z dysku części macierzy i wykonywanie obliczeń fragmentami.

W 1979 roku zastosowano układy pamięci 4K×1 bit, co umożliwiło powiększenie pamięci operacyjnej do 4 Msłów. Rozbudowano system wejścia/wyjścia, w którym cztery procesory obsługują 48 kanałów I/O. W ten sposób powstał **Cray-1s**. W tym samym czasie światło dzienne ujrzał **CYBER 205** o podobnej architekturze i parametrach, co **Cray**.

W kwietniu 1982 roku zaprezentowano **Cray X-MP** powstały przez połączenie dwóch jednostek centralnych **Cray-1** dzielących wspólną pamięć operacyjną o pojemności 4 Msłów. Istnieje również wersja superkomputera z czterema procesorami. Długość rejestrów wektorowych zwiększono do 512 słów. Zastosowano nowe układy pamięci o czasie dostępu 6 ns, cykl zegarowy skrócono do 9.5 nanosekund.

**Cray-2** zbudowany został dwa lata później. Składa się on z trójwymiarowych modułów całkowicie zanurzonych w chłodzącej cieczy fluorokarbonowej. Długość przewodów skrócono do 16 cali (41 centymetrów), dzięki czemu uzyskano długość cyklu zegarowego wynoszącą 4 ns. Komputer ten składa się z czterech procesorów, dysponuje PAO o pojemności 32 Msłów i jest dwunastokrotnie szybszy niż **Cray-1**.

Ulubionym zajęciem amerykańskich informatyków jest dzielenie mocy obliczeniowej urządzenia przez jego cenę. Stwierdzono, iż rezygnacja z wyrafinowanej technologii pozwoli na wielokrotne zmniejszenie kosztów komputera — większe, niż spadek jego mocy obliczeniowej. Według tej filozofii na początku lat osiemdziesiątych skonstruowano w firmie Convex superkomputer **C-1**. Składa się on z pamięci operacyjnej, podsystemów wejścia/wyjścia i jednostki centralnej, którą omówię dokładniej. Jednostka ta została zbudowana z pięciu procesorów mogących wykonywać swoje zadania równolegle (rys. 3). O wydajności maszyny decyduje procesor wektorowy złożony z dwóch arytmometrów wektorowych i jednostki kontrolnej.

Oba arytmometry (**VPU** — ang. *Vector Processor Unit*) są wyposażone w trzy procesory pracujące z mikrooprogramowaniem. Jeden odpowiada za zapełnianie rejestrów wektorowych danymi oraz przesyłanie wyników do pamięci, drugi zajmuje się dodawaniem wektorów oraz operacjami logicznymi, a trzeci — mnożeniem i dzieleniem. Mogą pracować potokowo wykonując równocześnie dodawanie i mnożenie. Wykonanie równolegle dwóch operacji dodawania lub dwóch mnożeń nie jest możliwe.

**VPU** mają po cztery akumulatory zawierające 128 liczb 64-bitowych każdy. Podobnie jak w systemie Cray istnieją 128-bitowe rejestry maski i 8-bitowe rejestry długości wektora.

Magistrale danych mają szerokość 64 bity plus 8 bitów kontroli parzystości. Arytmometry mogą operować na liczbach 32-bitowych pojedynczej precyzji korzystając równocześnie z tej samej magistrali. W tym przypadku jej linie zostają podzielone na „nieparzyste" i „parzyste". Jeden **VPU** korzysta z linii o numerach 1, 3, 5... a drugi 2, 4, 6...

Architektura **C-1** została zorientowana na pamięć operacyjną. Na początku cyklu obliczeń jednostka kontrolna procesora wektorowego pobiera z procesora instrukcji informacje o ilości danych do pobrania z pamięci i ich adresach. W czasie pobierania danych i wypełniania nimi rejestrów wektorowych przystępują do pracy specjalizowane podsystemy jednostki kontrolnej. Podsystem pobierania i przesyłu zajmuje się ustalaniem zawartości rejestrów maski, łączeniem i podziałem wektorów, przekazywaniem informacji pomiędzy rejestrami. Dysponuje własną pamięcią podręczną o pojemności 28 KB.

Podsystemy dodawania i operacji logicznych oraz mnożenia i dzielenia odpowiadają procesorom **VPU**. Są przeznaczone do przekazywania tym procesorom mikroprogramów, oba dysponują 24 KB pamięci podręcznej. Układ arbitrażu reguluje dostęp poszczególnych podsystemów do magistrali wewnętrznej procesora wektorowego i magistral zewnętrznych.

**C-1** został zbudowany z układów TTL Schottky'ego oraz układów scalonych wykonanych w technologii CMOS, zawierających po 8000 bramek logicznych. Rejestry wektorowe wykonano z ultraszybkich pamięci RAM w technologii MOS. Elementy komputera umieszczono na standardowych kartach o szerokości 19 cali. Komputer jest chłodzony powietrzem.

**Convex** został wyposażony w kompilator języka

Fortran 77, w którym napisano tysiące programów użytkowych z różnych dziedzin techniki. Kompilator automatycznie rozpoznaje części programu nadające się do przetwarzania potokowego i optymalizuje kod wynikowy tak, aby jak najwięcej operacji było wykonywanych równolegle. Jednak największą zaletą maszyny jest jej cena. Dwa lata temu wynosiła pół miliona dolarów — tyle, co dobrego mini-komputera; dziesiątą część ceny **Cray-1**. Tymczasem moc obliczeniowa **C-1** wynosi 6 MFLOPS, czwartą część mocy **Cray-1**.

Za pierwszy superkomputer, w którym w pełni wykorzystano zalety architektury wektorowej można uznać zbudowany na początku 1986 roku **Matrix I**. Sercem komputera jest zestaw do 32 procesorów potokowych, które mogą wykonywać równolegle operacje dla wielu elementów wektora. W zestawie może być maksymalnie 256 megabajtów pamięci RAM, przepustowość magistrali systemowej wynosi 320 MB/s. Jednostką peryferyjną maszyny może być mini-komputer **VAX**. Producent komputera opracował bibliotekę procedur w Fortranie realizujących obliczenia inżynierskie, naukowe i z dziedziny przetwarzania sygnałów. Model komputera w pełnej konfiguracji ma wydajność 1000 MFLOPS i kosztuje 4 miliony dolarów; wersję z ośmioma procesorami można nabyć już za 896 tysięcy dolarów.

W komputerach o dużej liczbie równolegle pracujących procesorów wydajność zależy od przepustowości łączącej je magistrali. Nie można uniknąć dokonywanej co pewien czas wymiany rezultatów cząstkowych, nazywanej synchronizacją obliczeń. Dla liczby procesorów większej niż kilkanaście przestaje wystarczać pojedyncza magistrala — jednostki muszą zbyt długo czekać na swoją kolejkę dostępu do niej.

Coraz częściej spotykane są konstrukcje, w których każdy procesor może się bezpośrednio połączyć z wieloma innymi. Jednostki są rozmieszczane wewnątrz sześcianu, stąd nazwa rozwiązania — Hypercube. Za przykład niech posłuży rodzina maszyn serii T firmy Floating Point Systems. Komputery te składają się z 16 do 4096 (na razie tylko teoretycznie) węzłów. W każdym umieszczono procesor Transputer, 64-bitowy koprocesor wektorowy, 1 megabajt pamięci dynamicznej RAM i 32-bitowy procesor nadzorczy wyposażony w 2 KB własnej pamięci statycznej. Z każdego węzła wychodzą cztery magistrale mogące obsługiwać po cztery kierunki. Dwa połączenia poświęcono na potrzeby systemu, dwa na operacje wejścia/wyjścia i komunikację z pamięcią masową — pozostało 12 na wymianę informacji między węzłami.



W chwili, gdy piszę te słowa, najszybszym na świecie superkomputerem jest ETA GF 10 wykonujący 10 miliardów operacji zmiennoprzecinkowych w ciągu sekundy. Pierwszy egzemplarz pojawił się w 1986 roku, wtedy maszyna kosztowała 20 milionów dolarów. Sercem komputera jest zespół kilku procesorów wektorowych chłodzonych ciekłym azotem. Zostały one wykonane w technologii CMOS, z układów dających opóźnienie sygnału 500 pikosekund na bramce logicznej. Pamięć operacyjna została wykonana na układach MOS, ma pojemność 256 megabajtów i jest chłodzona za pomocą układu kriogenicznego. Komunikację z otoczeniem zapewnia 16 kanałów we/wy o przepustowości 60 MB/s każdy.

Na koniec 1987 roku zapowiedziano prezentację komputera Cray-3 zbudowanego z układów ECL wykonanych z arsenku galu. Obecnie nie są jeszcze znane żadne bliższe dane o tej konstrukcji.

Budową superkomputerów zajmują się nie tylko firmy amerykańskie. Maszyny japońskich koncernów NEC, Hitachi i Fujitsu niewiele ustępują parametrami konkurentom zza Pacyfiku. Własne superkomputery budują również ZSRR i Chiny. Specjaliści oceniają, że udział superkomputerów w sprzedaży dużych maszyn w ciągu dziesięciu lat zwiększy się o połowę.

## ROZSTRZYGNIĘCIE KONKURSU NA PROGRAM EDUKACYJNY NA KOMPUTER MSX

Czas już ogłosić rezultaty konkursu na program edukacyjny na komputer pracujący w standardzie MSX, jaki rozpisaliśmy w numerze 2/88 „InforMika". Pod adresem redakcji nadesłano kilkanaście rozwiązań, z których 4 najlepsze jury konkursu w składzie: Jerzy Klawiński (przewodniczący), Jacek Nowicki (sekretarz), Roland Wacławek i Edward Krawczyński (członkowie) postanowiło uhonorować nagrodami. Jednocześnie jury zadecydowało nie przyznawać nagrody pierwszej, gdyż ogólny poziom nadesłanych rozwiązań nie był wysoki.

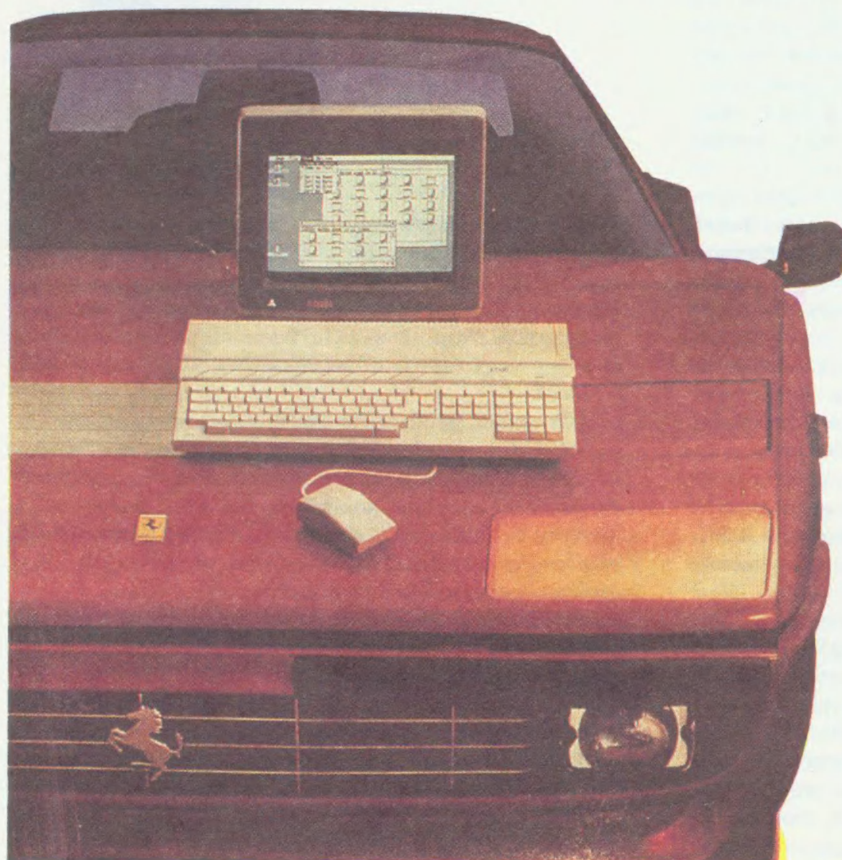
**Nagrodę drugą** otrzymał kol. **Mirosław Kwiatek** z Warszawy za program edukacyjny o charakterze gry dydaktycznej pt. „Okopy". Program ten służy do nauki rysunku maszynowego i ćwiczenia wyobraźni przestrzennej, a może zostać wykorzystany jako niezła pomoc naukowa szczególnie dla uczniów techników i szkół zawodowych.

**Nagrodę trzecią** przyznano kol. **Piotrowi Buczkowskiemu** z Mosiny i kol. **Arturowi Chodackiemu** z Żywca, autorom programu „Mapa". Program służy do nauki geografii Polski (rozmieszczenie większych miast). Wykorzystano w nim gotowy algorytm gry komputerowej — śmigłowca latającego nad określonym obszarem.

Jury przyznało również dwa **wyróżnienia**. Pierwsze z nich otrzymał kol. **Krzysztof Matela** z Bydgoszczy za program „Józef" stanowiący komputerowe kalendarium życia i działalności Józefa Piłsudskiego. Drugie wyróżnienie przypadło w udziale kol. **Stefanowi Nawrockiemu** z Koszalina za program „Bartek", będący dosyć ciekawą symulacją pracy dźwigu suwnicowego.

Nagrody w konkursie stanowiły zestawy dyskietek ufundowanych przez austriacką firmę Prosystem GMbH z Wiednia oraz książki o tematyce informatycznej ufundowane przez redakcję „InforMika". Wszystkim nagrodzonym i wyróżnionym gratulujemy i życzymy przyszłych sukcesów w pracy z komputerami.

**Jury konkursu**



ROLAND WACŁAWEK

# TURBO-PASCAL
## i zewnętrzne funkcje maszynowe

Wszystko wskazuje na to, że TURBO-Pascal 3.0 stał się w Polsce najpopularniejszym językiem programowania dla komputerów PC/XT zarówno wśród użytkowników profesjonalnych, jak i amatorów. Dla tych ostatnich istotny jest zwłaszcza fakt, że do użytkowania kompilatora wystarcza w zupełności system w minimalnej konfiguracji, z pojedynczą stacją dysków. Tak więc TURBO-Pascal 3.0 służy do zadań najprzeróżniejszych, poczynając od budowy banków danych a kończąc na oprogramowaniu systemowym. W tym ostatnim przypadku na ogół trzeba uzupełnić program pascalowy „wstawkami" maszynowymi.

Instrukcja **INLINE** nadaje się do tworzenia wstawek krótkich, w typowym przypadku liczących nie więcej niż kilka rozkazów. Tam, gdzie potrzebny jest nieco dłuższy program maszynowy, lepiej dołączyć go w gotowej postaci z zewnętrznego pliku dyskowego. Dodatkową zaletą tej metody jest możliwość uruchomienia i przetestowania programu maszynowego „poza" TURBO-Pascalem, korzystając w pełni z takich narzędzi jak makroassemblery, debuggery itd, bez konieczności kłopotliwej zmiany formy programu na bajtowy zapis **INLINE**. Nie ma przy tym ograniczeń na rozmiar dołączanego w ten sposób programu maszynowego. Jedyna restrykcja wynika z faktu, że sumaryczna objętość kodu maszynowego dołączonej procedury wraz z kodem programu pascalowego, wygenerowanym przez kompilator, nie może przekroczyć 64 KB.

Procedury i funkcje, przeznaczone do dołączenia z zewnątrz, muszą być przygotowane w postaci plików typu **.COM** lub **.BIN** (ich format jest identyczny), będących w rzeczywistości mapą pamięci operacyjnej. Podczas kompilacji programu pascalowego zewnętrzny program maszynowy jest konsolidowany z kodem maszynowym, wyprodukowanym przez kompilator.

Dołączony z zewnątrz program maszynowy jest traktowany jak samodzielna procedura lub funkcja. Deklaracja procedury lub funkcji zewnętrznej składa się

z samego nagłówka, uzupełnionego słowem: **EXTERNAL** z następującym po nim napisem, przedstawiającym nazwę pliku, zawierającego kod maszynowy procedury lub funkcji. Rozszerzenia **.COM** nie trzeba podawać. Jeżeli plik znajduje się w katalogu innym niż roboczy, trzeba nazwę pliku poprzedzić specyfikacją ścieżki. Przykład:

```
PROCEDURE Melodyjka; EXTERNAL 'MELODIA';
PROCEDURE Generuj_impulsy(Liczba, dlugosc, odstepinteger);
                        EXTERNAL 'C:\ASEMBLER\IMPULSY'
FUNCTION  Dlugosc_impulsu:Integer; EXTERNAL 'DL_IMPU.BIN';
```

Odwołanie do procedury lub funkcji zewnętrznej odbywa się tak samo, jak do pozostałych procedur lub funkcji.

Dołączona procedura maszynowa musi spełnić pewne warunki. Ponieważ nie sposób z góry przewidzieć lokalizacji programu zewnętrznego w pamięci operacyjnej, musi on być relokowalny (ściślej: niezależny od lokalizacji, ang. *position independent*) i nie może zawierać żadnych odwołań do segmentu danych chyba, że segment ten jest identyczny z segmentem kodu. Przykładowo, jeśli procedura zewnętrzna używa zmiennych roboczych, to powinny być one zlokalizowane w obszarze kodu albo na stosie procesora, a odwołania do nich powinny odbywać się za pośrednictwem rejestru **CS** lub **SS**, np. **MOV AX, CS**: dana albo **MOV BX, (BP 7)**. Aby uniknąć ciągłego stosowania w adresacji przedrostka CS:, można na początku procedury skopiować do rejestru DS zawartość CS, oczywiście po uprzednim przechowaniu pierwotnej zawartości DS, np. tak:

PUSH CS : wyślij zawartość CS na stos

POP DS : zdejmij wartość ze stosu, ładując ją do DS.

Jeżeli zmienne robocze procedury maszynowej mają mieć charakter statyczny, tzn. zachowywać wartość między jej kolejnymi wywołaniami, muszą być zlokalizowane w obszarze kodu procedury maszynowej. W chwili powrotu z wywołania zawartości rejestrów: **DS**, **ES**, **SS**, **SP** i **BP** muszą być takie same, jak w chwili wywołania.

Parametry aktualne (inaczej: rzeczywiste) są w TURBO-Pascalu 3.0 przekazywane procedurom i funkcjom za pośrednictwem stosu procesora. Dotyczy to także zewnętrznych procedur maszynowych. W chwili wywołania procedury zewnętrznej, na szczycie stosu znajduje się dwubajtowy adres powrotu. Parametry aktualne, jeśli występują, znajdują się poniżej adresu powrotu (pod kolejnymi, wyższymi adresami stosu). W przypadku funkcji, najniżej ułożonym parametrem są komórki zarezerwowane dla rezultatu funkcji.

Jeśli parametr jest przekazywany przez zmienną, parametr jest reprezentowany na stosie przez czterobajtowy wskaźnik zmiennej (segment: *offset*), będący faktycznie absolutnym adresem zmiennej w pamięci. Natomiast przy przekazywaniu parametrów przez wartość, na stosie jest lokowana wprost wartość parametru. Jej format jest taki sam, jak przedstawiono w rozdziale poświęconym wewnętrznej reprezentacji podstawowych typów danych w TURBO-Pascalu 3.0. Przypomnijmy je pokrótce:

Wartości typu **real** zajmują na stosie 6 bajtów. Pierwszy bajt to wykładnik, reszta tworzy mantysę, poczynając od bajtu najmłodszego do najstarszego. Mantysa jest zawsze znormalizowana, tzn. najstarszy bit piątego bajtu mantysy należy uważać za równy 1. Normalnie bit ten pełni jednak funkcję znaku mantysy (0 = liczba nieujemna, 1 = liczba ujemna). Wykładnik jest zapisywany z przesunięciem 80. (np. wartość 85 oznacza, że mantysę należy pomnożyć przez 2 do potęgi 5). Zerowa wartość bajtu wykładnika powoduje uznanie całej liczby za równą 0.

Wartości typu **integer** i jego typów okrojonych wykraczających poza przedział 0..255, jak również typów wyliczeniowych, liczących ponad 255 elementów są zapisywane w dwóch bajtach w kodzie uzupełnieniowym do 2 (U2). Bajt mniej znaczący znajduje się pod niższym adresem.

Wartości typu **byte**, **boolean** i **char**, typów okrojonych typu podstawowego **integer** o wartościach nie wykraczających poza przedział 0..255 i typów wyliczeniowych, obejmujących nie więcej niż 256 elementów, są normalnie zapisywane w pojedynczym bajcie. Przy przekazywaniu danych na stosie wartości te zajmują jednak na zawsze 2 bajty (drugi bajt jest nie wykorzystany). Elementy typów porządkowych są reprezentowane przez ich numery porządkowe. Numeracja zaczyna się od 0 (np. *false* = 0, *true* = 1).

Łańcuch (**STRING**) zajmuje na stosie tyle bajtów, ile wynosi jego maksymalna (zadeklarowana) długość, powiększona o 1. Pierwszy bajt zawiera bieżącą długość łańcucha, następne bajty przechowują kolejne znaki.

Wartości typu zbiorowego (**SET**) zajmują na stosie zawsze 32 bajty (przy przechowywaniu zmiennych obszar ten może być mniejszy w zależności od liczebności zbioru). Numer bitu w bajcie, reprezentującego dany element zbioru, można określić ze wzoru: **nr__porz MOD 8**, zaś przesunięcie zawierającego ten bit bajtu względem pierwszego bajtu zbioru wynosi **(nr__porz DIV 8)—(min DIV 8)**, gdzie nr__porz jest numerem porządkowym rozważanego elementu zbioru w typie podstawowym.

Wskaźniki są zapisywane w pamięci za pomocą czwórki bajtów. Pierwsze dwa bajty zawierają przesunięcie adresowe zmiennej wskazywanej, dwa ostatnie — adres bazowy (segment). Wartość NIL jest reprezentowana przez czwórkę bajtów o wartości 0.

Zamiast wartości tablic i rekordów, na stosie jest umieszczany czterobajtowy wskaźnik pierwszego bajtu tablicy lub rekordu (segment: *offset*).

Wywołanie funkcji zewnętrznej następuje rozkazem **CALL NEAR**. Powrót z funkcji lub procedury zewnętrznej najprościej zrealizować więc rozkazem maszynowym **RET NEAR**. Jeśli program zewnętrzny modyfikuje zawartość rejestrów **CS**, **DS**, **ES** i **BP**, to powinien je na początku przechować, a przed powrotem odtworzyć. Jeżeli do procedury lub funkcji przekazywano parametry, należy je przed powrotem z procedury usunąć ze stosu. Najprościej przeprowadzić to rozkazem **RET** *nn*, gdzie *nn* — liczba bajtów do usunięcia, równa liczbie bajtów zajętej na stosie przez parametry.

Wywołanie funkcji zewnętrznej różni się od wywołania procedury tym, że przed umieszczeniem na stosie parametrów, TURBO-Pascal 3.0 rezerwuje na nim miejsce dla wartości funkcji — tyle bajtów, ile zajmuje wartość typu, odpowiadającego typowi funkcji (**char:** 1, **integer:** 2, **real:** 6, **STRING** [8]: 9, itd.). Zarezerwowany obszar niekoniecznie musi jednak uczestniczyć w przekazywaniu wartości. Warto podkreślić, że o ile

przy przekazywaniu parametrów typy jednobajtowe zawsze zajmują dwa bajty, to przy rezerwacji miejsca na wynik — tylko 1 bajt. Pole wartości dla funkcji typu **SET** zajmuje zawsze 32 bajty.

Sposób przekazywania wartości (rezultatu) funkcji zewnętrznej zależy od typu funkcji. Wartości typów porządkowych, reprezentowane przez numer porządkowy, są przekazywane w rejestrze **AX**. Jeśli rezultat jest jednobajtowy, należy przy tym wyzerować rejestr **AH**. Wyjątkiem są funkcje typu **boolean**, przekazujące wynik za pośrednictwem flagi procesora ZF. Flaga ZF = 1 (zero) reprezentuje wartość *false*, ZF = 0 (nie zero) — *true*. Wszelkie wskaźniki są przekazywane za pośrednictwem pary rejestrów **DS:AX**. Jeśli wartości są przekazywane przez rejestry lub flagi, to obowiązkiem programu maszynowego jest przed powrotem usunąć ze stosu także i bajty, zarezerwowane dla wartości funkcji, lecz nie wykorzystane.

Wartości typu *real*, **SET** i **STRING** muszą w chwili powrotu znajdować się na wierzchołku stosu, w zarezerwowanym dla nich miejscu, w odpowiednim formacie wewnętrznym. W przypadku funkcji typu *Real* wystarczy po prostu nie usuwać ze stosu komórek, przeznaczonych na wynik.

Przykład: funkcja zewnętrzna **suma__XY** jest typu **integer** i oblicza sumę dwóch parametrów, także typu **integer**. Kod maszynowy funkcji jest zawarty w pliku: **SUMA__XY.BIN**. Program oczekuje na podanie dwóch liczb całkowitych, oddzielonych spacją:

```
PROGRAM  Przyklad_funkcji_EXTERNAL;
VAR      alfa, beta: integer;

  FUNCTION suma_xy(a, b:integer):integer;
        EXTERNAL 'SUMA_XY.BIN';

BEGIN
  Write('Podaj dwie liczby calkowite: ');
  Readln(alfa, beta);
  Writeln(suma_xy(alfa, beta))
END.
```

Oto stan wierzchołka stosu systemowego w chwili uruchomienia procedury maszynowej po wywołaniu funkcji **suma ab**:

| | |
|---|---|
| Miejsce na wartość funkcji | SP+6 |
| Wartość pierwszego argumentu | SP+4 |
| Wartość drugiego argumentu | SP+2 |
| Adres powrotu z wywołania | SP |

Oto postać źródłowa programu maszynowego **SUMA__XY**. Pierwszą czynnością po wywołaniu jest zapamiętanie pierwotnej zawartości rejestru **BP** (zawartość SP wzrasta przy tym o 2) i skopiowanie **SP** do **BP**. W ten sposób adres **[BP:4]** odpowiada ostatniemu argumentowi, **[BP:6]** przedostatniemu, itd. Miejsce zarezerwowane na wartość funkcji pozostanie w tym przypadku nie wykorzystane, gdyż wartości typu integer są przekazywane przez rejestr **AX**. W związku z tym program maszynowy musi usunąć ze stosu nie tylko cztery bajty, zarezerwowane dla parametrów, ale i dwa następne, przeznaczone dla wartości:

```
Suma_xy:  PUSH  BP           ;przechowaj na stosie zawartosc BP
          MOV   BP, SP       ;skopiuj do BP aktualny stan SP
          MOV   AX, [BP+4]   ;AX := wartość drugiego parametru
          ADD   AX, [BP+6]   ;dodaj do AX pierwszy parametr
          POP   BP           ;odtworz zawartosc rejestru BP
          RET   6            ;powrot i usunięcie param. ze stosu
```

Oto przykład jeszcze jednej funkcji maszynowej: **CZYTZNAK**, podającej kod ekranowy znaku umieszczonego we wskazanym wierszu i kolumnie ekranu (w przypadku kart grafiki barwnej dotyczy to zawsze strony 0). Funkcja jest typu char, jej obydwa argumenty typu **integer**. Oto ogólny schemat jej wywołania:

```
PROCEDURE Czytzn(kol,wiersz:integer):integer;
        EXTERNAL 'CZYTZNAK'
```

Oto przykład użycia funkcji zewnętrznej **CZYTZNAK**. Najpierw odczytuje ona zawartość 24 górnych linii ekranu i przechowuje w tablicy, a następnie wyprowadza zarówno wiersze, jak i znaki w odwrotnym porządku:

```
PROGRAM Odczyt_ekranu;

VAR     linia, kol : integer;
        znak       : ARRAY[1..80, 1..24] OF char;

  FUNCTION czytznak(kolumna, wiersz: integer) : char;
                                  EXTERNAL 'CZYTZNAK';
BEGIN FOR linia:= 1 TO 24 DO
        FOR kol:= 1 TO 80  DO
            znak[kol, linia]:= czytznak(kol, linia);
      ClrScr;
      FOR linia:= 24 DOWNTO 1 DO
        FOR kol:= 80 DOWNTO 1 DO Write(znak[kol, linia]);
      REPEAT UNTIL KeyPressed
END.
```

Jak działa funkcja maszynowa **CZYTZNAK**? Oto jej asemblerowa postać źródłowa (w przypadku karty monochromatycznej wartość stałej **Segm__Ekr** należy zmienić na **OBOOOII**):

```
          TITLE CZYTZNAK
Segm_Ekr  EQU    0B800H      ;adres segmentowy pamięci ekranu CGA
          ORG 100H

Czytaj_zn SEGMENT
          ASSUME CS:Czytaj_Zn

Poczatek: PUSH BP            ;przechowaj na stosie zawartość BP
          MOV  BP, SP        ;skopiuj do BP aktualny stan SP
          PUSH DS            ;zapamiętaj na stosie zawartość DS
          MOV  BX, Segm_Ekr  ;wpisz do rejestru segmentowego DS
          MOV  DS, BX        ;adres segmentowy pamięci ekranu
          MOV  AL, [BP+4]    ;AL:=młodszy bajt drugiego parametru
          DEC  AL            ;skoryguj numerację z 1..80 na 0..79
          MOV  AH, 80        ;AH:= liczba znaków w wierszu ekranu
          MUL  AH            ;wyznacz numer 1 znaku w tym wierszu
          MOV  BX, AX        ;skopiuj nr pierwszego znaku do BX
          MOV  AL, [BP+6]    ;AL:= młodszy bajt pierwszego param.
          DEC  AL            ;skoryguj numeracje z 1..25 na 0..24
          XOR  AH, AH        ;wyzeruj starszy bajt rejestru AX
          ADD  BX, AX        ;oblicz w BX efektywny numer znaku
          ADD  BX, BX        ;BX:= offset znaku w pamięci ekranu
          MOV  AL, [BX]      ;pobierz kod znaku do rejestru AL
          POP  DS            ;odtworz zawartość rejestru DS
          POP  BP            ;odtworz zawartość rejestru BP
          RET  5             ;powrót i usunięcie param. ze stosu

Czytaj_Zn ENDS
          END
```

Ponieważ w TURBO-Pascalu numeracja wierszy i kolumn ekranu w trybie tekstowym zaczyna się nie od 0, lecz od 1, jest potrzebna wstępna korekcja współrzędnych. Dla prostoty nie przewidziano synchronizacji dostępu do pamięci z odchylaniem wiązki elektronów, ale nie zawsze jest to potrzebne, np. w przypadku kart monochromatycznych lub sporadycznych odczytów przy karcie CCA. Oto budowa wierzchołka stosu po wywołaniu:

| | |
|---|---|
| Miejsce na zwartość typu char | SP:5 |
| Wartość pierwszego argumentu | SP:4 |
| Wartość drugiego argumentu | SP:2 |
| Adres powrotu z wywołania | SP |

Ponieważ tym razem funkcja jest typu char, na jej wartość rezerwuje się na stosie nie dwa bajty, lecz tylko jeden. Ostatnim rozkazem maszynowym jest w związku z tym **RET 5**, powodujący usunięcie ze stosu 5 zarezerwowanych uprzednio bajtów.

Standardowy makroasembler, np. MASM, dostarcza plików typu **.OBJ**, które należy poddać konsolidacji za pomocą programu konsolidatora (ang. — *linker*), np. LINK. Konsolidator produkuje pliki typu **.EXE**. Aby uzyskać plik typu **.BIN** lub **.COM**, należy posłużyć się programem EXE2BIN.

*Najszybsza z prezentowanych drukarek 24-igłowych — FUJITSU DL 5600. Normalna prędkość wynosi do 485 znaków na sekundę!*

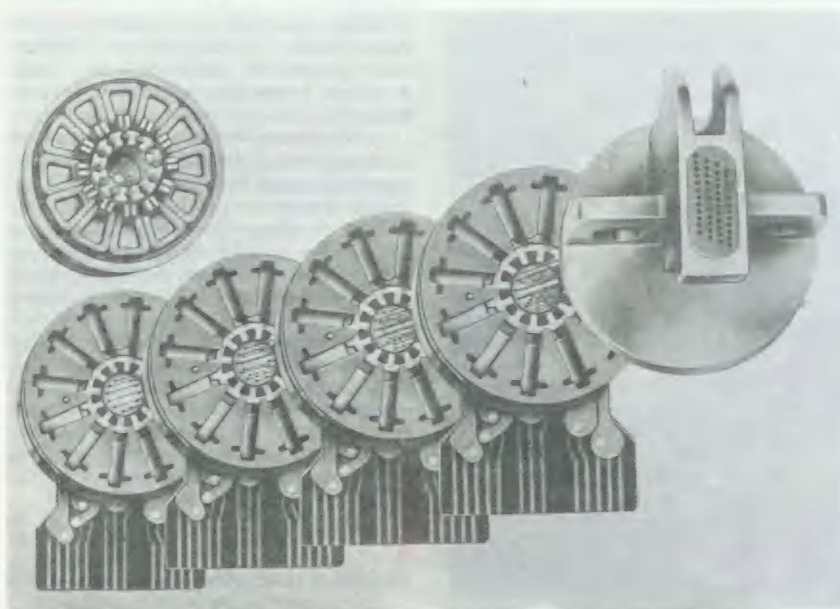


# DRUKARKI '88

**TADEUSZ RZEPECKI**

Od początku ery mikrokomputerowej spośród wszystkich peryferyjnych urządzeń najwięcej zmian obserwuje się w zakresie różnych typów drukarek. Główne nowości roku 1988 zostały przedstawione na marcowych targach CeBIT w Hanowerze. Nadal trwa rywalizacja pomiędzy głównymi producentami oferującymi sprzęt coraz lepszy, szybszy i bardziej wytrzymały. Kilka lat temu, po wprowadzeniu drukarek laserowych, wydawało się, iż ten kierunek zastąpi najlepsze rozwiązania drukarek mozaikowych. Jednakże zbyt wysoka cena oraz dalszy rozwój drukarek igłowych zmienił te tendencje. Tegoroczne rozwiązanie drukarki 48-igłowej jest praktycznym potwierdzeniem dalszych perspektyw drukarek mozaikowych. Model EPSONA przedstawiony na targach (na razie bez symbolu, sprzedaż od lipca 1988 roku) zapewnił jakość druku nie gorszą niż drukarka laserowa tej samej firmy. Tego typu rozwiązanie zapewnia ogromną gęstość punktów (360×360 punktów na cal) niemożliwą dotąd do uzyskania bez użycia drukarek laserowych. Minimalna odległość między kolejnymi punktami wynosi 0.065 mm, co w praktyce oznacza znaczne pokrycie się sąsiednich punktów. Matryca znaków w trybie o największej rozdzielczości z odległości najlepszego widzenia (ok. 30 cm) jest praktycznie odczuwana jako ciągła linia z lekko rozmytymi konturami znaku, charakterystycznymi także dla drukarek laserowych. Ilustruje to najlepiej porównanie matrycy normalnej linii na trzech najpopularniejszych drukarkach — 9-, 24- i 48-igłowej. Oczywiście zastosowanie tak dużej ilości igieł możliwe było w 4 rzędach z przesunięciem o 1/4 odległości w pionie następnego rzędu. Powstał również problem sterowania ruchem igieł w czasie pracy oraz odprowadzenia ciepła z głowicy drukującej. Na zamieszczonym zdjęciu pokazano szczegóły konstrukcyjne głowicy tej drukarki — widoczne są poszczególne segmenty odpowiadające za ruch poszczególnych rzędów igieł. Dodatkowo pokazany został zespół elektromagnesów wprawiających w ruch ostatni zestaw igieł. Dla przykładu zamieszczamy próbkę możliwości otrzymania różnych krojów pisma przy pomocy drukarki 48-igłowej. Prędkość druku przy stosowaniu trybu LQ wynosi 100 znaków/sekundę, w trybie normalnej pracy 300 znaków na sekundę. Drukarka ta stanowi poważne zagrożenie cenowe dla drukarek laserowych, których cena oscyluje w granicach 5—10 tysięcy DM. Natomiast kolejnym skokiem cenowym w dół firma EPSON wprowadziła na rynek kolejną drukarkę 24-igłową. Przewiduje się, że jej cena ma osiągnąć 1000 DM. Drukarka o symbolu LQ-2550 posiada możliwość druku w 7 kolorach przy maksymalnej gęstości w trybie graficznym 360×360 punktów na cal. Prędkość druku jest też interesująca — 400 znaków na sekundę przy gęstości 12 znaków na cal w trybie normalnym i 133 znaki w trybie LQ. Daje to imponujące prędkości odpowiednio 13.7 s/stronę standardowego maszynopisu i 23.8 s/stronę w trybie LQ.

*Głowica drukarki 48-igłowej. W lewym górnym rogu przedstawiono zespół elektromagnetyczny jednego rzędu igieł*



Dla amatorów firma EPSON proponuje małą i zgrabną drukarkę 9-igłową z możliwością druku w 7 kolorach. Drukarka EX-800 umożliwia druk w 2 wariantach trybu NLQ, w trybie normalnej pracy jej szybkość wynosi 300 znaków na sekundę. Znaczna gęstość druku w poziomie (do 240 punktów na cal) umożliwia stosowanie wielu krojów liter, często o znacznym zagęszczeniu.

Dla profesjonalistów wymagających bardzo szybkiego pisania ta sama firma proponuje model DFX-5000 za około 4000 DM. Cena staje się uzasadniona w momencie określenia trwałości głowicy drukującej na około 100 milionów znaków oraz maksymalnej szybkości druku 533 znaki na sekundę przy matrycy znaku 9×7 punktów. Maksymalna szerokość papieru używanego do druku wynosi 406.6 mm, możliwy jest druk w oryginale i 5 kopiach. Jedna ka-




*Jeden z przedstawicieli klasy drukarek laserowych — BROTHER HL-8 LaserAs*




*Bogatą ofertę przedstawiła firma OKI. Na zdjęciu 24-igłowa drukarka ML 391*

seta bez wymiany wystarcza na zapisanie 15 milionów znaków (około 17 000 stron maszynopisu) przy średniej ilości 14 punktów na znak. Maksymalny czas druku 1 strony standardowego maszynopisu wynosi 7.4 sekundy, a w trybie LQ/NLQ 24.4 sekundy. Użycie tej drukarki jest uzasadnione w przypadkach ośrodków obliczeniowych przy bardzo dużej ilości drukowanych informacji typu wykazów, tabel, obliczeń, list osobowych itp.

Nie bez powodu tak wiele miejsca poświęciliśmy firmie EPSON, przedstawiła ona bowiem najszerszą ofertę dla wszystkich odbiorców w pełnej gamie parametrów szybkości, jakości druku, możliwości graficznych i kolorystyki swoich drukarek. W tym wyścigu firma STAR pozostała chwilowo nieco z tyłu. Jej oferta następcy NL 10 — drukarka o symbolu LC 10 lub LC 10C (Commodore) z kolorową wersją odpowiada mniej więcej modelowi EPSON EX-800. Na przykładzie tej drukarki można jednak zaobserwować tendencje w budowie głowic. Trwałość i szybkość elementów mechanicznych zależy przede wszystkim od sił bezwładności generowanych przez poruszającą się głowicę drukującą. Postęp w tej dziedzinie widać na przykładzie głowicy drukarki NL 10 i LC 10. Około dziesięciokrotne zmniejszenie masy głowicy doprowadziło do zwiększenia trwałości, a także obniżenia poziomu hałasu i wibracji w czasie druku, co jest bardzo istotnym parametrem przy długim przebywaniu w otoczeniu każdej drukarki.

Ciekawą ofertę przedstawiła firma OKI. Proponuje ona serię drukarek 9-, 18- i 24-igłowych o symbolach odpowiednio MICROLINE 190, 290, 390. Szczególnie ciekawe są drukarki ML 192 i ML 193. Bardzo małe rozmiary (ML 192 — 360×275×80 mm) oraz bateryjne zasilanie z możliwością zasilania akumulatorowego 12 lub 24 V przy małej masie (4.5 kg) pozwalają na zastosowanie tych drukarek w warunkach podróży, szczególnie do współpracy z mikrokomputerami przenośnymi, walizkowymi, ostatnio bardzo modnymi. Wbudowany akumulator umożliwia pracę ciągłą w czasie 1 godziny lub przerywaną do 4 godzin.

Najszybszą drukarkę 24-igłową zaprezentowała firma FUJITSU. Model DL 5600 umożliwia zapis do 485 znaków na sekundę, a w trybie LQ 162 znaków na sekundę. Drukarka umożliwia drukowanie w 7 kolorach, a także emulację pracy wszystkich najbardziej znanych drukarek z możliwością dostosowania do każdego systemu PC.

Ugruntowaną już pozycję mają drukarki laserowe, chociaż ich cena nie ulega zmianie. Zagrożeniem dla ich popularności mogą stać się dopiero drukarki 48-igłowe. Prawie każda firma oferowała co najmniej jeden model drukarki laserowej, graniczna



*LQ-2550 to nowa propozycja bardzo szybkiej i taniej drukarki 24-igłowej*

*LC 10 w wersji kolorowej — tania drukarka znanej firmy STAR*

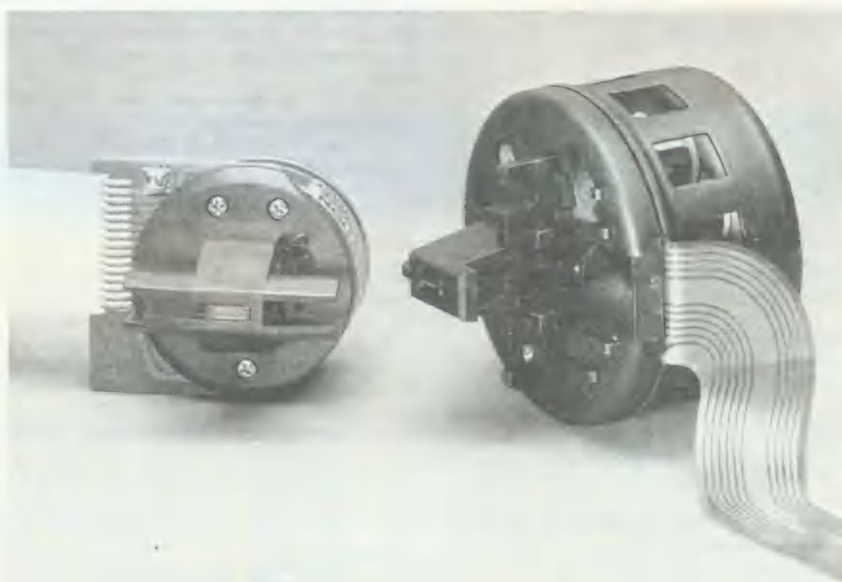



prędkość tych drukarek wynosi jednak 8 do 10 stron na minutę — wynika ona z zasady druku. Typowe przykłady takich drukarek to CI-5 firmy C.ITOH oraz HL-8 LaserAs firmy Brother. Cyfra przy symbolu określa prędkość w stronach na minutę. Powszechnie używany jest laser półprzewodnikowy, typowa pojemność bufora pamięci wynosi od 512 KB do 3 MB. Oczywiście każda drukarka tego typu posiada opcję pracy zgodną z HP LaserJet oraz EPSON FX 80 jako przyjętego standardu.

Ponieważ jednak do tej pory istniała dosyć wyraźna różnica w prędkościach popularnych drukarek, nawet tak szybkich jak laserowe, a typową poligrafią, firma C.ITOH postarała się o wypełnienie tej luki. Dla firm wymagających bardzo szybkiego i częstego kopiowania przygotowanych tekstów opracowano model drukarki jonowej o symbolu CIE 3000. Ten nowy sposób przenoszenia i utrwalania obrazu zapewnia znaczne przyspieszenie druku do 45 stron na minutę w przypadku modelu CIE 3000 L4. Maksymalna rozdzielczość jest wystarczająca dla zapewnienia bardzo dobrej jakości druku (300×300 punktów na cal). Drukarka ta umożliwia przy normalnej pracy i optymalnym wykorzystaniu jej czasu pracy wykonanie 150 tysięcy stron w ciągu miesiąca, przy trwałości elementów mechanicznych około 5 milionów stron. Z pewnością istnieje zatem możliwość drukowania za jej pomocą np. materiałów z sympozjów, wyciągów z prac naukowych, fragmentów danych statystycznych itp.

Istnieje jednak grupa odbiorców, którym nie odpowiada forma znaków z drukarek punktowych. Firma Brother przeznacza właśnie dla nich swoje drukarki, do których bardziej pasuje określenie: elektroniczne maszyny do pisania. Dzięki zastosowaniu wyświetlacza ciekłokrystalicznego umożliwiają one kontrolę tekstu znajdującego się w buforze. Niektóre modele umożliwiają pracę z zasilaniem bateryjnym, co stanowi dużą motywację do zakupu przez piszących na „zielonej trawce". Wprawdzie prędkość druku nie jest imponująca (ok. 40 znaków na sekundę) lecz jakość przy zastosowaniu taśmy węglowej może zadowolić najwybredniejszych. Ceny zróżnicowane w pełnym asortymencie — od 400 DM dla prostej drukarki termicznej, poprzez kilkaset DM dla popularnych typów, do 4000 DM w przypadku pełnego zestawu do obróbki tekstów z wbudowaną stacją dysków 3.5 cala — EM 2000.

Przedstawione powyżej ogólne tendencje w ofertach nowych drukarek zmierzają do uzupełnienia luk w jakości druku, szybkości, optymalnej cenie przy zamierzonych parametrach i stworzenia możliwości wyboru przez klienta odpowiedniego wyrobu. Na pewno rywalizujące firmy nie powiedziały jeszcze ostatniego słowa, czego dowodzi tegoroczna oferta.



*Prace nad zwiększeniem trwałości doprowadziły do wielokrotnego zmniejszenia rozmiarów głowicy drukarki. Na zdjęciu głowice drukarek LC 10 (z prawej)*




*48-igłowa drukarka firmy EPSON — największa z niespodzianek w tej dziedzinie*

MARCIN KRECZMER

# PROTOKÓŁ TRANSMISJI INTERFEJSU TIMEX RS-232

W sklepach Centralnej Składnicy Harcerskiej, za złotówki, oraz w Baltonie, za dewizy, można kupić interfejs TIMEX RS-232 służący do połączenia ZX Spectrum lub TIMEX-a 2048 z dowolną drukarką wyposażoną w złącze szeregowe RS-232. W skład kompletu wchodzi kabel i kaseta z programem umożliwiającym transmisję danych pomiędzy komputerem i drukarką.

```
*** Listing 1 ***

 1         ORG  65100
 2
 3 ;***inicjacja programu***
 4         LD   HL,#A19E
 5         LD   (#5C24),HL
 6         RET
 7
 8 ;***rekord kanału***
 9         DEFW 65112 ;wyj.
10         DEFW 5572  ;wej.
11         DEFB #52   ;nazwa
12
13 ;***rozpoznawanie kodów**
14 START   CP   #17
15         JR   NZ,LFE60
16         LD   A,#09
17         JR   DRUK
18 LFE60   CP   #80
19         JR   C,DRUK
20         CP   #A5
21         JR   NC,LFE6C
22         LD   A,#3F
23         JR   DRUK
24 LFE6C   SUB  #A5
25         CALL #0C10
26         RET
27
28 ;***przesłanie bajtu ***
29 DRUK    PUSH AF
30 LFE73   CALL #1F54
31         JR   C,LFE7D
32         POP  AF
33         RST  8
34         DEFB #0C
35         JR   KONIEC
36 LFE7D   IN   A,(#BF)
37         RRA
38         JR   C,LFE73
39         DI
40         LD   A,#0F
41         OUT  (#BF),A
42         CALL PETLA
43         POP  DE
44         LD   E,#08
45 LFE8D   RR   D
46         JR   C,LFE97
47         LD   A,#0F
48         OUT  (#BF),A
49         JR   LFE9B
50 LFE97   NOP
51         XOR  A
52         OUT  (#BF),A
53 LFE9B   CALL PETLA
54         DEC  E
55         JR   NZ,LFE8D
56         XOR  A
57         OUT  (#BF),A
58         CALL PETLA
59 KONIEC  EI
60         RET
61
62 ;***pętla czasowa***
63 PETLA   PUSH BC
64         LD   BC,#D901
65 PAUZA   DJNZ PAUZA
66         DEC  C
67         JR   NZ,PAUZA
68         POP  BC
69         RET
```

Najwięcej zastrzeżeń w tym zestawie budzi program na kasecie. Wykorzystuje on do transmisji danych kanał 7. Z tego powodu nie działają instrukcje LLIST i LPRINT (w celu otrzymania wydruku trzeba używać instrukcji PRINT #7, LIST #7) i co za tym idzie, nie można uzyskać wydruków z wszystkich programów wykorzystujących kanał 3, standardowo przeznaczony dla drukarek. Program jest nierelokowalny (adres startowy 65100) co często uniemożliwia wykorzystanie go wraz z np. edytorem lub innym programem zajmującym ten sam obszar pamięci.

Proponuję dokonanie zmian mających zaradzić jego niedostatkom.

Listing 1 prezentuje program z kasety po deasemblacji. Dla uzyskania czytelności programu zamieniono ważniejsze etykiety z wygenerowanych przez deasembler (typu LADR) na symboliczne i wpisano pseudoinstrukcje typu DEFW. Dopisano też komentarze.

```
*** Listing 2 ***

 10 LET sk=0
 20 FOR n=65100 TO 65207
 30 READ a: POKE n,a: LET sk=sk+a
 40 NEXT n
 50 IF sk=11739 THEN  SAVE "rs 232"CODE 65100,108: STOP
 60 PRINT  FLASH 1;"blad w liniach DATA": LIST 100
100 DATA 33,197,92,54,225,35,54,233,205
102 DATA 197,92,17,8,0,25,34,197,92,201
104 DATA 254,23,32,4,62,9,24,18,254,128
106 DATA 56,14,254,165,48,4,62,63,24,6
108 DATA 214,165,205,16,12,201,197,245,205,84
110 DATA 31,56,5,241,207,12,24,49,219,191
112 DATA 31,56,241,243,62,15,211,191,6,216
114 DATA 205,60,5,209,30,8,203,26,56,6
116 DATA 62,15,211,191,24,4,0,175,211,191
118 DATA 6,216,205,60,5,29,32,234,175,211
120 DATA 191,6,216,205,60,5,193,251,201
```

Przyjrzyjmy się najpierw procedurze inicjacji programu. Do rejestru HL ładowany jest adres względny rekordu kanału (linia 4). Jest to liczba, jaką trzeba dodać do zmiennej wskazywanej przez CHANS (zwykle 23734), aby otrzymać adres, pod jakim w pamięci komputera znajduje się rekord kanału. Adresy względne wszystkich dołączonych kanałów są przechowywane w zmiennej systemowej STRMS. Następnie zawartość rejestru HL jest ładowana do komórek STRMS opisujących kanał 7 (linia 5). Zmiana adresu w linii 5 na adres 23580 = #5C1C odpowiadający kanałowi 3, pozwoli na swobodne operowanie instrukcjami LLIST i LPRINT. W programie rekord kanału jest zdefiniowany od adresu ORG+8, a procedura wyjścia, czyli właściwy program przesyłania danych do drukarki znajduje się pod adresem ORG+12 (etykieta START). Stąd, aby dokonać asemblacji pod inny adres należałoby: zmienić argument pseudoinstrukcji ORG, w linii 4 wpisać wartość odpowiadającą ORG+8—23734 i w linii 9 wpisać adres równy ORG+12. Nie spieszmy się jednak!

Ponieważ kanał 7 nie jest kanałem standardowym musiał zostać w programie zdefiniowany. Natomiast kanał 3, przeznaczony do obsługi drukarki, jest w Spectrum zdefiniowany począwszy od adresu 23749. Stąd nasz program możemy znacznie uprościć, wystarczy zamienić adres procedury wyjścia na adres odpowiadający etykiecie START. Kasujemy wszystkie linie od 4 do 11 i na ich miejsce wpisujemy:

```
4     LD HL, START
5     LD (23749), HL
6     RET
```

Tak zapisany program możemy umieścić w dowolnym obszarze pamięci zmieniając tylko wartość argumentów „adr" pseudoinstrukcji ORG i inicjować przez RANDOMIZE USR adr.

Należy w tym miejscu zwrócić uwagę na to, że zarówno zmienna CHANS jak i adres względny kanału 3 może być programowo zmieniony. W takim wypadku procedurę inicjalizacji trzeba by odpowiednio przekształcić. Praktycznie takiej konieczności nie spotkałem, nie przewiduje jej także program oryginalny.

Nie jest to koniec naszych kłopotów. Program będzie działał znakomicie, dopóki będzie wpisany pod adresy większe niż 32767 czyli w górną połowę pamięci. Winna temu jest pętla czasowa umieszczona w końcowej części programu i specjalizowany procesor WE WY ULA, który ma wyższy priorytet w dostępie do obszaru pamięci RAM z zakresu #4000 ÷ #7FFF w celu odczytu obrazu i wstrzymuje pracę procesora. Wyjaśnijmy to dokładniej.

Drukarka wymaga określonej szybkości transmisji danych. Najczęściej spotykaną szybkością transmisji jest 1200 bodów i taką prędkość przewiduje nasz program. W tym celu w programie po wysłaniu każdego bitu za pomocą instrukcji OUT, następuje skok do pętli czasowej. Pętla zapewnia niezbędną zwłokę czasową równą 1/1200 sekundy. Pętla zbudowana jest w taki sposób, by łatwo można było uzyskać inne szybkości transmisji. Np. dla dwukrotnie wolniejszej wystarczy w linii 64 wpisać do rejestru C liczbę 2 H (LD BC #D902). Nasz problem polega jednak na tym, że aby program działał prawidłowo, pętla czasowa (a przynajmniej instrukcja DJNZ), nie może znajdować się w obszarze pamięci RAM z zakresu #4000 ÷ #7FFF.

Przyjmując standardową prędkość przesyłu danych — 1200 bodów — możemy ten fragment programu uprościć do:

```
63      PETLA PUSH BC
64            LD B, #D9
65 STOP DJNZ STOP
66            POP BC
67            RET
```

Pozostaje problem umieszczenia linii 65 poza dolną połową pamięci. Nie byłoby wygodnym rozwiązaniem wpisywanie jej osobno w inne miejsce pamięci. Instrukcji w tej linii odpowiadają kody #10 i #FE. W pamięci ROM znajdujemy sekwencję #10, #FE, #C9 odpowiadającą kodom rozkazu szukanego i instrukcji RET. Wystarczy więc zamienić linię 65 na:

```
65            CALL #053C
```

i program jest gotowy do asemblacji.

Sądzę, że dla osób posługujących się choć trochę asemblerem, posiadanie takiego programu źródłowego może być cenne chociażby ze względu na możliwość dokonywania dalszych zmian. Np. do edytorów tekstu czy programów drukujących grafikę nie będą potrzebne linie rozpoznawania kodów i usuwając je można program znacznie skrócić.

Listing 2, zapisany w BASIC-u daje kod wynikowy jeszcze wygodniejszy w użyciu — w pełni relokowalny. Wystarczy przepisać program i gdy nie będzie błędów w liniach DATA, ukaże się komunikat *Start tape, then press any key*. Program nagrany w ten sposób na taśmę możemy wgrać pod dowolny adres komendą LOAD „rs 232" CODE adr i uruchomić przez RANDOMIZE USR adr.

Zmiany dokonane w tym programie w stosunku do omówionych poprzednio są już niewielkie. Ograniczają się do wpisania pętli czasowej bezpośrednio po każdej instrukcji OUT, a nie jako osobnego podprogramu i dopisania na początku programu kilku instrukcji umożliwiających programowi rozpoznanie, w jakim miejscu pamięci się znajduje. Sposób realizacji powyższego pozostawiamy dociekliwości Czytelników.

# CIEKAWE KSIĄŻKI

*Jan Bielecki, SAM NA SAM Z JĘZYKIEM C, Wydawnictwa Komunikacji i Łączności 1988, nakład 50 000 egz. Książka, ma podtytuł „ZX Spectrum" i jest adresowana (zgodnie ze słowami Autora) do użytkowników komputerów ZX Spectrum i Elwro 800 Junior, którzy chcieliby opanować język C, a których przygotowanie informatyczne niewiele wykracza poza umiejętność posługiwania się językiem BASIC. W założeniu książka stanowi kompletny podręcznik opisujący podstawy języka C, ilustrowany przykładami możliwymi do zrealizowania za pomocą implementacji Hisoft na komputerze ZX Spectrum (lub zgodnym z nim).*

*Jan Bielecki jest znanym specjalistą w zakresie języków programowania wyższego rzędu; w szczególności jest autorem wielu książek i publikacji traktujących o języku C. Rzetelność, stanowiąca cechę charakterystyczną dla poprzednich jego publikacji powoduje, że książkę tę kupujemy z nadzieją na dobry podręcznik dla początkujących. Niestety, spotyka nas rozczarowanie.*




*Dostępny na komputerach ZX Spectrum (i zgodnych z nim) kompilator firmy Hisoft udostępnia użytkownikowi podzbiór instrukcji języka C oraz pewne niestandardowe rozszerzenia, będące konsekwencją architektury komputera. Część rozdziałów stanowi szczegółowy opis tego podzbioru. Jego związek z systemem ZX Spectrum jest jednak bardzo luźny i sprowadza się jedynie do uwag i odnośników. Pozostałe rozdziały poświęcone są problemom związanym z architekturą ZX Spectrum (potraktowanym, niestety, pobieżnie).*

*Styl Autora — jak we wszystkich jego publikacjach — jest bardzo zwięzły. Wydaje się jednak, że w tego typu pracy stanowi to wadę; początkujący użytkownik kompilatora Hisoft C będzie miał duże kłopoty z „przegryzieniem się" przez tak podany materiał.*

*Wobec części ściśle związanej z ZX Spectrum można postawić zarzut niekompletności (książka ukazała się w pierwszej połowie kwietnia 1988 r. i w tym też czasie pisana była ta recenzja; dostępność instrukcji do opisywanego kompilatora sprowadzała się praktycznie do trudno osiągalnych na giełdzie kserokopii). W szczególności:*

- *opisana lista dyrektyw preprocesora nie jest kompletna (nie uwzględniono dyrektyw #list, #direct, #error);*
- *opis wykonywania dyrektyw wejścia-wyjścia jest niepełny (m.in. brak wskazań plikowych do wszystkich urządzeń wejścia-wyjścia), odczuwa się brak opisu praktycznego sposobu wykorzystania magnetofonu jako pamięci zewnętrznej, opisu formatu plików na taśmie (różnych od formatu systemu ZX Spectrum), itp.;*
- *lista znaków kodu ASCII powinna być szerzej opisana;*
- *lista błędów kompilatora jest opisana w sposób krańcowo oszczędny — może należało pokusić się o dosłowne przetłumaczenie opisu błędów z anglojęzycznej instrukcji, gdzie każdemu błędowi poświęcone jest przynajmniej kilkanaście linii?*

*Podczas posługiwania się książką odczuwa się brak skorowidza rzeczowego.*

*Autor pominął też milczeniem błędy implementacji kompilatora Hisoft C, które niekiedy w sposób dokuczliwy dają znać o sobie.*

*Osobliwy wydaje się być również dobór danych tekstowych stosowanych w przykładach konstrukcji języka C. Część z nich pisana jest w języku polskim (np. „Pierwszy program"), a część w angielskim (np. „JanB is 44 now"), i to bez uzasadnionych przyczyn. Według mnie należało się zdecydować na jeden z tych języków i trzymać się go konsekwentnie.*

*Reasumując: jeżeli dysponujemy pozycjami „Język C" autorstwa B. W. Kernighana i D. M. Ritchie (tłum. D. i M. Kruszewskich) wspomnianymi przez Autora we wstępie i spisie literatury oraz fabryczną instrukcją kompilatora, to nie mamy po co kupować tej książki, bo nie dowiemy się z niej nic nowego. Jeżeli tak nie jest, możemy ją zakupić, ale (parafrazując) jest to praktyczna implementacja przysłowia „na bezrybiu i rak ryba".*

*Książka wydana jest starannie, na dobrej klasy papierze i za to wydawnictwu należy się uznanie. Nie ustrzeżono się jednak błędów (np. na str. 26 w definicji struktury OutStr zamienione są wiersze 9 i 10, na str. 65 w wierszu 5 zamieniono kropkę z przecinkiem). Erraty nie załączono. Zastanawiająca jest także kompozycja graficzna okładki (autorstwa p. Lucjana Madziara). Trudno domyślić się konkretnego związku okładki z tematyką książki. Przypomina się znakomity rysunek (o ile pamiętam autorstwa amerykańskiego rysownika Chasa Adamsa) przedstawiający grupę osób kontemplującą wiszący na ścianie abstrakcyjny obraz składający się z chaotycznie rozmieszczonych figur geometrycznych. Jedna z nich stwierdzała: „A może autor obrazu nie chciał nam przez to NIC powiedzieć...". Może nie chciał.*

**(dap)**



DARIUSZ A. PRZYGODA

# PRZYSPIESZONA WSPÓŁPRACA Z MAGNETOFONEM

Poniższy artykuł ma na celu zilustrowanie na wybranym przykładzie sposobu usunięcia kilku niedogodności powstających przy współpracy komputera ZX Spectrum z magnetofonem.

Z perspektywy czasu można powiedzieć, że w minionych kilku latach powstała duża ilość ciekawego oprogramowania o charakterze narzędziowym i użytkowym napisanego na ten komputer (oczywiście biorąc pod uwagę jego ograniczone możliwości i przestarzałą konstrukcję). Autorzy tego oprogramowania starali się w miarę możliwości tuszować wady komputera, i niekiedy udawało im się to w zadziwiający sposób.

Jedną z takich niedogodności powoduje protokół współpracy komputera z nośnikami pamięci zewnętrznej. Jak wiadomo, do składowania programów ZX Spectrum wykorzystuje dowolny magnetofon. (Istnieją oczywiście: szybka pamięć kasetowa — *microdrive* i wiele typów stacji dysków elastycznych, jednak z powszechnie znanych powodów założymy, że użytkownik dysponuje jedynie magnetofonem). Informacja pomiędzy urządzeniami przesyłana jest szeregowo, zgodnie z niestandardowym protokołem, a prędkość przesyłu informacji wynosi średnio 1500 bodów (średnio — bo inny jest czas przesyłania logicznego „0" a inny „1"). Dobór takich parametrów transmisji zapewnił jej dużą niezawodność, a co za tym idzie, umożliwił pewne uniezależnienie się od egzemplarza magnetofonu, na którym program był nagrywany. Niestety — ceną był długi czas wczytywania (nagrywania) danych.

Rozważmy problem przyspieszenia transmisji danych. Przeprowadzone (przy nie zmienionym protokole transmisji) próby wykazują, że:

- **dwukrotne przyspieszenie transmisji** jest na dobrej jakości sprzęcie w zasadzie nieodczuwalne (nie ma kłopotów z wczytywaniem programów przy użyciu sprzętu innego niż użyty do nagrywania), na sprzęcie klasy popularnej problemy te występują sporadycznie,
- **trzykrotne przyspieszenie transmisji** jest możliwe do zrealizowania jedynie na sprzęcie wysokiej klasy, przy czym występują problemy z wczytywaniem programów nagrywanych na innym sprzęcie,
- duży wpływ na zapewnienie prawidłowości transmisji ma **jakość kasety** (test dostępnych kaset krajowych i zagranicznych wykazał znaczny wpływ własności mechanicznych kasety),
- **trwałość zbiorów** utworzonych przy dwukrotnym przyspieszeniu transmisji (przy przesterowaniu nagrania do poziomu +6 dB) nie różni się od trwałości zbiorów utworzonych normalnie.

Z powyższych rozważań wynika, że dwukrotne skrócenie czasu transmisji jest absolutnie możliwe, co szczególnie gdy chodzi o programy używane często, jest znaczną oszczędnością czasu.

Naprzeciw tym potrzebom wyszła w 1983 roku firma Ness Micro Systems, wypuszczając popularny program Speedyload. Jest to napisany w BASIC-u generator programu w kodzie maszynowym (1330 bajtów), który po umieszczeniu w górnej połówce pamięci umożliwia wykorzystywanie w BASIC-u nowego zestawu operacji do współpracy z magnetofonem — z dwukrotnie większą szybkością transmisji. Pewną wadą programu jest jego nierelokowalność — umożliwia on wygenerowanie kodu pod jednym ze ściśle określonych 30 adresów. Dokładniejsza analiza kodu wykazuje, że jest to (z dokładnością do kilkudziesięciu pierwszych bajtów) odpowiednik procedur współpracy z magnetofonem zawartych w pamięci ROM komputera, a jedyne różnice sprowadzają się do innych stałych czasowych. Program ten stanowi doskonały „półprodukt", który można zainstalować w wielu używanych przez siebie programach, a czas wydatkowany na przeróbkę zwróci się z nawiązką!

Nie sposób podać uniwersalnej recepty na przeróbkę programu, gdyż każdy ma inną strukturę, jednakże opisana poniżej przeróbka bazy danych MASTERFILE powinna naświetlić problem, umożliwiając owocną działalność w przerabianiu programów we własnym zakresie.

Baza danych MASTERFILE (napisana w 1982 roku przez programistów firmy CAMPBELL SYSTEMS) jest obecnie jedną z najbardziej popularnych baz danych dostępnych na tym komputerze. Poniżej podane zostały cechy programu, które są istotne podczas przeróbki programu.

**Struktura programu:** dwa bloki, jeden w języku BASIC (pełniący funkcje programu ładującego, zbioru podprogramów obsługi urządzeń peryferyjnych jak magnetofon czy drukarka oraz bufora rekordów danych umieszczonych w obszarze zmiennych programu), drugi w kodzie maszynowym (zajmujący lokacje począwszy od 57328), stanowiący właściwy kod bazy danych.

**Sposób zapamiętywania danych:** możliwość nagrania jedynie rekordów (jako tablicy znakowej), lub całego programu (jako ciągu: samouruchamiający się program z obszarem zmiennych programowych + blok bajtów stanowiący kod maszynowy bazy danych).

Istnienie drugiej z tych cech nakłada na nas pewne warunki do spełnienia. Program MASTERFILE nagrywany jako oddzielna całość musi zostać nagrany w formie umożliwiającej ponowne wczytanie do komputera bez konieczności ładowania programu Speedyload. Przykładowa sekwencja bloków danych generowanych podczas procesu zgrywania programu jest następująca:

1. krótki program ładujący w BASIC-u,
2. blok bajtów zawierający procedury szybkiego ładowania,
3. program w BASIC-u (wraz z obszarem zmiennych programowych) stanowiący część bazy danych napisanej w BASIC-u i rekordy danych,
4. blok bajtów stanowiący właściwy kod bazy danych.

Bloki 1 i 2 zgrywane są z normalną prędkością transmisji, natomiast 3 i 4 z prędkością podwójną.

Pozornie trudny problem generowania dwóch różnych programów w BASIC-u podczas jednego procesu zgrywania rozwiązać można na wiele sposobów. W opisywanym przykładzie program (1) stanowi jedną linię (o numerze 9999) programu (3). Linia ta jest normalnie nie używana przez program (3) podczas jego normalnej pracy i służy jedynie do ustawienia zmiennej RAMTOP, wczytania kodu programu Speedyload i zapewnienia dalszego procesu ładowania ze zwiększoną prędkością. Specjalna kilkunastobajtowa procedura SVL (dołączona do bloku (2)) powoduje zgranie na tej jednej linii wyposażonej w nagłówek umożliwiający autostart programu. Procedura ta (listing 1) wyposażona jest w tak dużą ilość komentarzy, że zbędne jest przytaczanie jej dokładniejszego opisu.

Różnice w strukturach programów: zmodyfikowanego i nie zmodyfikowanego podane są w tabeli. Poniżej zamieszczony jest dokładny opis postępowania umożliwiającego uzyskanie programu MASTERFILE wyposażonego w procedury Speedyload. Jako oprogramowanie narzędziowe służące do dokonania przeróbki wykorzystano monitor MONS3M21 i asembler GENS3M21 (oba są produktami firmy Hisoft).

TABELA

| ADRES | MASTERFILE | MASTERFILE PRZEROBIONY |
|---|---|---|
| (PROG) | Początek programu w BASIC-u | Początek programu w BASIC-u |
| (zależnie od programu w BASIC-u) | - | Linia 9999 stanowiąca procedurę ładowania MASTERFILE |
| (VARS) | Początek obszaru zmiennych programowych (rekordy danych) | Początek obszaru zmiennych programowych (rekordy danych) |
| 55973 | - | RAMTOP |
| 55974 | - | Procedura SVL |
| 56033 | - | Procedura SPD |
| 57327 | RAMTOP | |
| 57328 | Początek kodu MASTERFILE | Początek kodu MASTERFILE |
| 65535 | Koniec obszaru RAM | Koniec obszaru RAM |

1. Do wyzerowanego komputera wgrywamy program Speedyload. W odpowiedzi na pytanie o adres początku tworzonego bloku podajemy parametr 30 (co spowoduje przyjęcie adresu 33316).
2. Usuwamy Speedyload instrukcją NEW (nie powoduje to zniszczenia wygenerowanego kodu, gdyż Speedyload wykonuje instrukcję CLEAR 32767) i ustawiamy RAMTOP na adres 23999 (instrukcją CLEAR 23999).
3. Wgrywamy asembler GENS3M21 od adresu 35000 i monitor MONS3M21 od adresu 24000.
4. Uruchamiamy asembler GENS3M21 i sprawdzamy (za pomocą dyrektywy X) jaki jest początek bufora tekstu źródłowego (dla podanych wartości powinien on być równy 43866). Następnie powracamy do BASIC-a (dyrektywą B).
5. Uruchamiamy monitor MONS3M21. Dokonujemy (dyrektywą T) deasemblacji kodu wynikowego Speedyload tak, aby utworzony tekst źródłowy zaczynał się od adresu 43866, czyli początku bufora tekstu źródłowego asemblera. W tym celu specyfikujemy następujące parametry (wartości dla danego przykładu podano w nawiasach):
   — adres początkowy obszaru deasemblowanego (33316, czyli #8224);
   — adres końcowy obszaru deasemblowanego (33316 + 1330, czyli #8756);
   — adres początkowy utworzonego kodu źródłowego (43866, czyli #AB5A);
   — adres początkowy obszaru roboczego monitora (ten wybieramy opcjonalnie, np. #F000).

Po dokonaniu deasemblacji monitor poinformuje nas o adresie końca utworzonego tekstu źródłowego (53040, czyli #CF30). Adres ten musimy umieścić w komórkach pamięci, w których asembler GENS3M21 przechowuje adres końca tekstu źródłowego (są to 54 i 55 komórka za adresem początku programu). W ten

LISTNG 1

```
SVL     ORG     55974 ;
        LD      HL, 9998        ; numer linii - w liniach 9998 i
                                ; 9999 mieści się program ładujący
        CALL    #196E           ; procedura LINE_ADDR zwraca w HL
                                ; adres początku wyspecyfikowanej
                                ; linii
        PUSH    HL              ; adres początku linii na stos
        LD      HL, (#5C4B)     ; do HL adres końca programu
                                ; wyznaczony zmienną VARS
        POP     BC              ; adres początku linii do BC
        AND     A               ; CY := 0
        SBC     HL, BC          ; w HL długość linii 9998 i 9999
        LD      (LEN), HL       ; długość do obszaru nagłówka
        LD      (PRG), HL       ; j.w.
        PUSH    HL              ; długość na stos
        PUSH    BC              ; początek linii 9998 na stos
        LD      IX, HDR         ; do IX adres początku obszaru
                                ; nagłówka
        LD      DE, 17          ; do DE długość nagłówka
        XOR     A               ; CY := 0 (sygnał "będzie zrzucany
                                ; nagłówek")
        CALL    #04C2           ; procedura SAVE_BYTES
        POP     IX              ; odtworzenie adresu startu
        POP     DE              ; odtworzenie długości
        LD      A, #FF          ; sygnał "będzie zrzucany blok
                                ; bajtów"
        CALL    #04C2           , procedura SAVE_BYTES
        RET                     ; powrót do systemu

HDR     DEFB    0               ; znak "program"
NAME    DEFM    "KATALOG   ";   nazwa programu (10 bajtów)
LEN     DEFB    0, 0            ; długość kodu programu
STR     DEFB    9998?256        ; adres linii startowej (9998)
        DEFB    9998/256
PRG     DEFB    0, 0            ; długość programu bez zmiennych
                                ; koniec listingu
```

LISTING 2

```
   1 GO TO USR VAL "58285"
4000 INPUT  PAPER VAL "7";(c$( TO VAL "32")); LINE c$:
     GO TO USR R
4010 COPY : GO TO USR R
4020 GO SUB VAL "9010": PRINT USR VAL "56033":
     SAVE c$( TO VAL "10") DATA f$(): GO TO USR R
4030 GO SUB VAL "9010": RANDOMIZE USR VAL "55974":
     POKE VAL "23736",VAL "187":
     SAVE "SPD"CODE VAL "55974",VAL "1354"
4031 PRINT USR VAL "56033":
     SAVE c$( TO VAL "10") LINE VAL "4035"
4032 PRINT USR VAL "56033":
     SAVE "MF mcode"CODE VAL "57328",VAL "8208": GO TO USR R
4035 PRINT USR VAL "56033": LOAD ""CODE : GO TO VAL "1"
4040 LET c$=STR$ (VAL c$( TO VAL "14")/VAL c$(VAL "15" TO VAL "19")):
     GO TO USR R
4050 PRINT USR VAL "56033": LOAD c$( TO VAL "10") DATA f$():
     GO TO USR R
7000 LET c$="": GO TO USR R
9000 BEEP SGN PI,NOT PI: GO TO USR R
9010 PRINT #0;"Start tape ...": PAUSE NOT PI:
     RANDOMIZE USR VAL "3438": RETURN
9999 CLEAR VAL "55973": LOAD "SPD"CODE :
     PRINT USR VAL "56033": LOAD ""
```

sposób asembler „zaakceptuje" wpisany przez nas w jego bufor tekst. Najprościej jest to wykonać z poziomu monitora wpisując (w danym przypadku) w komórkę o adresie 35054 wartość #30, a w komórkę o adresie 35055 wartość #CF. Po zrobieniu tego wracamy do BASIC-a dyrektywą <EDIT>.

6. Ponownie uruchamiamy asembler GENS3M21 (za pomocą tzw. ciepłego startu!). Dokonujemy następujących operacji na tekście źródłowym:
   — usuwamy wszystkie instrukcje NOP na końcu programu (tzn. po instrukcji JP L8267) — powodem ich występowania jest zaokrąglenie długości kodu wynikowego do 1330 B;
   — dodajemy pierwszą linię tekstu źródłowego o treści
   ORG 56033
   (spowoduje ona ulokowanie kodu od tego adresu); (**UWAGA!** W tym momencie dysponujemy odzyskaną z kodu wynikowego źródłową wersją programu Speedyload — warto ją nagrać na taśmę do późniejszego wykorzystania!)
   — usuwamy z tekstu źródłowego następujące sekwencje instrukcji:

```
        XOR   A
        LD    DE, #09A1
        CALL  #OCOA
        SET   5, (IY + 2)
        CALL  #15D4
```

   (odpowiedzialną za komunikat *Start tape...*) oraz

```
        LD    B, # 19
L8736   HALT
        DJNZ  L8736
```

   (odpowiedzialną za odstęp między nagłówkiem i ciałem bloku).
   Tak otrzymany tekst źródłowy asemblujemy (koniec kodu powinien wypadać na adresie 57327) i zgrywamy na taśmę dyrektywą O,,SPD.
7. Usuwamy tekst źródłowy z bufora asemblera (np. dyrektywą D) i wpisujemy program podany na listingu 1 (komentarze można pominąć). Asemblujemy tekst (koniec kodu powinien wypadać na adresie 56032) i zgrywamy tekst na taśmę dyrektywą O,,SVL. „Zaszytą" w programie nazwę KATALOG można zmienić na dowolną, pamiętając jedynie, aby miała ona długość dokładnie 10 B (w razie potrzeby dopełnić spacjami).
8. Zerujemy komputer i wykonujemy instrukcję CLEAR 55973.
9. Wygrywamy program MASTERFILE i przechodzimy do poziomu BASIC-a. Następnie wprowadzamy zmiany do części programu MASTERFILE napisanej w BASIC-u doprowadzając ją do postaci przedstawionej na listingu 2.
   Podczas dokonywania zmian należy uważać, aby nie zniszczyć zmiennych programowych BASIC-a (np. instrukcją CLEAR). Program MASTERFILE przechowuje rekordy danych i swoje dane wewnętrzne w postaci tablicy znakowej f$ (o niestandardowym formacie), i zniszczenie jej uniemożliwi prawidłową pracę programu. Jednocześnie należy unikać wprowadzania dodatkowych zmiennych (np. instrukcją LET); każda nowo utworzona zmienna zajmuje miejsce w pamięci uszczuplając obszar przewidziany na rekordy danych.
10. Po dokonaniu zmian wgrywamy wcześniej utworzone bloki SPD i SVL, a następnie uruchamiamy MASTERFILE instrukcją GO TO O (nie RUN!) i zgrywamy na taśmę dyrektywą V (z użyciem opcji P), co da nam samostartującą wersję programu wyposażonego w procedury szybkiego ładowania.

# i naj-NOWE nowsze



## Wieści z sympozjum GaAs

Co roku w Stanach Zjednoczonych organizowane jest sympozjum firm i uniwersytetów zajmujących się rozwijaniem technologii arsenku galu. Na ostatnim, które odbyło się w Portland, zaprezentowano najnowsze osiągnięcia z tej dziedziny. Dwaj naukowcy z RFN przedstawili układ mogący odbierać mikrofale o częstotliwości 35 GHz — dotąd skonstruowanie odbiorników o częstościach ponad 30 GHz uważano za niemożliwe. Firmy TriQuint i Raytheon zademonstrowały konwertery cyfrowo-analogowe i analogowo-cyfrowe pracujące z częstotliwością ponad 1 GHz.

Największe zainteresowanie wzbudził jednak Dave Kiefer z firmy Cray Research, który zrelacjonował stan prac nad superkomputerem Cray-3. Komputer będzie się składał z 16 procesorów o łącznej mocy obliczeniowej 12 000 MFLOPS. Wyprodukowano około stu rodzajów układów średniej skali integracji (MSI — zawierających trzysta do pięciuset bramek logicznych), z których składany jest Cray-3. Dzięki zastosowaniu technologii GaAs udało się zmniejszyć opóźnienie sygnału na bramce do 80 pikosekund (wobec 350 ps osiąganych w technologii ECL), co umożliwiło zastosowanie zegara systemowego o częstotliwości 500 MHz (wobec 244 MHz dla Craya-2s). Przewidziano niemal dwukrotne zwiększenie gęstości upakowania układów.

## Multisync XL

Firma NEC wyprodukowała monitor kolorowy o przekątnej ekranu 20 cali i maksymalnej rozdzielczości 1024×768 punktów. Monitor został wyposażony w układy automatycznie dostosowujące częstotliwość odchylania pionowego (w granicach od 50 do 90 Hz) i poziomego (od 21.8 do 50 kHz) do stosowanej karty graficznej. Można wyświetlać do 64 barw obrazu generowanego przez karty z wyjściem TTL (jak np. EGA) lub nieograniczoną ilość barw dla kart z wyjściem analogowym (jak PGA, VGA i MCGA). Producent zapewnia, że Multisync XL oprócz kart graficznych IBM PC/XT/AT może również współpracować z kartami graficznymi stosowanymi w komputerze Macintosh II.

## CLIPPER i Intergraph

Dotychczasowy producent mikroprocesora CLIPPER, firma Fairchild, połączyła się z produkującą własną rodzinę mikroprocesorów 32-bitowych firmą National Semiconductor. Wobec tego twórcy CLIPPER-a przeszli do firmy Intergraph tworząc nowy jej oddział — Advanced Processor Division. Zapowiedzieli nową wersję mikroprocesora — C300, (całkowicie zgodną z poprzednią C100) taktowaną częstotliwością 50 MHz, większą niż w poprzednim modelu, w którym wynosiła ona 33 MHz.

Nowa wersja mikroprocesora zostanie wykorzystana w produkowanych przez Intergraph stacjach roboczych (ang. *workstation*) rodziny InterAct, budowanych obecnie w oparciu o C100. Największy model rodziny (ang. *top of line*) jest wyposażony w 80 MB pamięci RAM, stację dysków 5,25 cala o pojemności 1,2 MB, dysk twardy o pojemności 156 MB i dwa kolorowe monitory o rozdzielczości 1184×844 punkty. Generacją obrazu zajmuje się procesor graficzny GX wyświetlający 512 kolorów z palety 16,7 miliona, kreślący obrazy z szybkością 100 tysięcy wektorów na sekundę.

Wydajność mini-komputera może być zwiększona za pomocą 64-bitowego procesora Floating Point Engine zbudowanego na bazie procesora wektorowego WEITEK 264/65 i układów innych producentów. FPE jest wyposażony we własną pamięć operacyjną o pojemności 8 MB oraz 1 MB pamięci do przechowywania mikroprogramów. Wydajność FPE (22 MFLOPS) nie ustępuje mocy superkomputera Cray-1.

Maszyny InterAct używają programów pracujących pod kontrolą systemu UNIX przeniesionych z mini-komputerów VAX oraz programów pracujących pod kontrolą MS-DOS. Możliwa jest współpraca z sieciami Ethernet, XNS, TCP/IP oraz emulacja terminali VT220, Tektronix 4107 i IBM 3270.

## IBM PC/AT wydajniejszy od PS/2

Firma Chips and Technologies oferuje zestaw czterech układów, które mogą zostać wykorzystane do budowy komputera klasy AT taktowanego częstotliwością 16 MHz. Są to: kontroler magistrali 82C-211, układ wspomagający system stronicowania pamięci 82C-212, kontroler peryferyjny 82C-206 i bufor adresów oraz danych 82C-215. Wszystkie układy są wykonane w technologii CMOS, dostarczane w obudowach z 84 wyprowadzeniami, mogą (zależnie od wersji) pracować z maksymalną częstotliwością 12 lub 16 MHz. Nie wywołują żadnych zakłóceń w pracy programów wykonywanych pod kontrolą MS-DOS czy OS/2 ani w działaniu kart rozszerzających stosowanych w IBM PC/AT.

Kontroler 82C-212 może zarządzać pamięcią RAM o pojemności od 1 do 8 MB optymalizując jej wykorzystanie. Podczas inicjalizacji pracy systemu zawartość BIOS jest automatycznie przepisywana do znajdujących się pod tymi samymi adresami bloków pamięci operacyjnej, co przyspiesza wykonywanie procedur systemowych. Podczas pracy z układami RAM o czasie dostępu 100 ns przy częstotliwości zegara 16 MHz kontroler wprowadza średnio 0,7 cyklu wyczekiwania.

## Jednopłytkowy sterownik

Sterownik przemysłowy SBS-1200 firmy Octagon Systems jest wyposażony w wykonany w technologii CMOS odpowiednik mikroprocesora Z80 taktowany częstotliwością 8 MHz. System zawiera 28 lub 92 KB pamięci statycznej RAM, kalendarz z zasilaniem bateryjnym i wbudowany programator EPROM/EEPROM. Użytkownik może dołączyć układy EPROM, EEPROM lub RAM z zasilaniem bateryjnym do przechowywania programów o długości do 44 KB. SBS-1200 zawiera interpreter języka ComBASIC (144 instrukcje, w tym 37 do obsługi komunikacji ze sterowanymi urządzeniami) oraz kompilator i debugger. System można programować za pomocą dołączonej klawiatury lub skrośnie — wykorzystując IBM PC i program Smartlink.

SBS-1200 wyposażono w trzy 16-bitowe liczniki i 32 linie we/wy, z których 24 mogą być programowo połączone w trzy dwukierunkowe kanały 8-bitowe. Przewidziano też dwa kanały RS-232 i złącze dla układów rozszerzających.

## Matryce logiczne

Matrycami logicznymi (ang. *gate arrays*) nazywamy układy VLSI realizujące złożone funkcje, pełniące rolę pomocniczą wobec mikroprocesora. Do nich należy np. stosowana w Spectrum ULA. Obecnie są na świecie produkowane układy scalone o nie zdefiniowanej strukturze wewnętrznej, którą odbiorca może ustalić sam poprzez niszczenie nadmiarowych połączeń między bramkami.

Dotąd prymat w technologii wytwarzania matryc logicznych dzierżyła firma Toshiba. Układ TC 110 G wykonany w technologii CMOS zawiera 50 000 bramek, szerokość ścieżek wynosi 1.5 mikrometra a opóźnienie na bramce — 700 pikosekund.

Powszechnie uważano, że skonstruowanie układu zawierającego 100 tysięcy bramek będzie możliwe dopiero po 1990 roku. Tymczasem już wiosną tego roku amerykańska firma LSI Logic zademonstrowała taki układ wykonany w technologii HCMOS o szerokości ścieżek 0,7 mikrometra. Układ zawiera 344 kanały we/wy. Producent twierdzi, iż jego układ może przetwarzać dane z wydajnością odpowiadającą mocy obliczeniowej mini-komputera VAX 11/780.

*Opracował* **Adam Nowicki**




JERZY GRĘDZIAK
ROLAND WACŁAWEK

# TOP DRIVE – sposób na modernizację stacji dysków ATARI 1050

Wielu Czytelników ma nam za złe, że nie poświęcamy w „InforMiku" należnej uwagi komputerom typu ATARI XL, które stanowią dziś — głównie dzięki działalności przedsiębiorstwa PEWEX — znaczący procent sprzętu w rękach polskich amatorów. Stan ten nie wynika jednak z naszej antypatii do tego komputera, lecz z faktu, że cieszy się on wprawdzie opinią doskonałego narzędzia do gier, ale nieco gorszego — do zastosowań praktycznych. Taki obraz komputerów ATARI XL kształtuje zwłaszcza dostępny osprzęt, stanowiący — z wyjątkiem joysticków — jego główną słabość.

Trzeba przyznać, że obok dobrego sterownika graficznego i generatora dźwięku ATARI XL ma zupełnie nieudany system interfejsów. Sposób pracy magnetofonu eliminuje proste sposoby podwyższenia gęstości zapisu w rodzaju TURBO TAPE dla C64. Standardowa stacja dysków 1050 jest, delikatnie mówiąc, niedopracowana, co dotyczy zarówno małej pojemności dysku (ok. 126 KB), jak i dość wolnej transmisji między komputerem, a dyskiem (19 200 bodów). Brak standardowego interfejsu drukarki skazuje z kolei użytkownika na kaleką, firmową drukareczkę, produkującą siermiężne znaki złożone z siatki 7×6 punktów przy ubogim zestawie atrybutów pisma. Jej możliwości są tak skromne, że wiele programów, zwłaszcza graficznych, żąda typowej drukarki np. standardu EPSON. Jej dołączenie wymaga jednak posiadania osobnego, dość drogiego interfejsu Centronics.

Wobec powyższych mankamentów panuje opinia, że w takich zastosowaniach, jak małe banki danych albo nawet redagowanie tekstów, ATARI XL jest mniej odpowiedni niż inne komputery podobnej klasy, a eliminacja wymienionych niedogodności wiąże się z dużymi kosztami i nie zawsze daje dobre wyniki. Opinię tę podzielaliśmy i my. Piszemy: podzielaliśmy, gdyż opinię tę zmodyfikowało niepozorne urządzenie o nazwie: TOP DRIVE, które zostało udostępnione nam do przetestowania przez firmę ATASERW z Nowej Dęby (woj. tarnobrzeskie).

Co to jest TOP DRIVE? Otóż jest to niewielka płytka z kilkoma układami scalonymi, wśród których znajduje się pamięć EPROM o pojemności 8 KB. Płytka ta jest wstawiana w cokół firmowej pamięci ROM na płycie montażowej stacji. Oryginalną pamięć ROM należy usunąć. Przestaje ona być potrzebna, gdyż funkcje zawartego w niej oprogramowania przejmują programy zawarte w pamięci EPROM na płytce.

Co daje TOP DRIVE? Przede wszystkim około pięciokrotne przyspieszenie transmisji między stacją, a komputerem: z 19 200 na ok. 70 000 bodów. Skorzystanie z przyspieszonej transmisji możliwe jest jednak tylko w przypadku programów zapisanych na dyskietce sformatowanej we właściwym dla TOP DRIVE formacie o podwójnej gęstości. Dzięki TOP DRIVE stacja 1050 pracuje w interesującym i bardzo popularnym na Zachodzie standardzie PERCOM, umożliwiającym m.in. automatyczne rozpoznawanie formatu dyskietki, programowe przełączanie gęstości zapisu i odczytu, niezależne formatowanie poszczególnych ścieżek i tworzenie nietypowych formatów.

Podwojona została też prędkość pozycjonowania głowicy stacji, co daje nie tylko bardzo istotne skrócenie średniego czasu dostępu (ważne zwłaszcza dla użytkowników baz danych), ale i oszczędza stację dysków. Jak to możliwe? Otóż głowica jest przemieszczana ze ścieżki na ścieżkę silnikiem krokowym, zaś każdy silnik ma swoją optymalną częstotliwość kroków. Jeśli częstotliwość ta jest za niska, to głowica podlega zbędnemu hamowaniu po zakończeniu każdego kroku, a następnie — ponownemu przyspieszeniu. Ruch na większe odległości nie jest więc płynny, co z jednej strony powoduje przykry hałas, z drugiej zaś niepotrzebne obciążenia elementów mechanicznych stacji. Częstotliwość krokowa w oryginalnej pamięci ROM stacji 1050 została dopasowana do przestarzałych typów napędów, tymczasem obecnie montowane są napędy nowszych typów, ze zwiększoną częstotliwością krokową.

Oprócz powyższych usprawnień, dzięki zmienionemu, doskonalszemu sposobowi formatowania, pojemność dyskietek rośnie o ok. 50 KB i wynosi 180 KB. Stacja 1050 bez TOP DRIVE może korzystać z formatu standardowego (720 sektorów po 128 bajtów) lub zoptymalizowanego (1040 sektorów po 128 bajtów). TOP DRIVE pozwala dodatkowo korzystać z formatu o podwójnej gęstości (720 sektorów po 256 bajtów). Niezależnie od tego wraz z TOP DRIVE jest dostarczany zes-

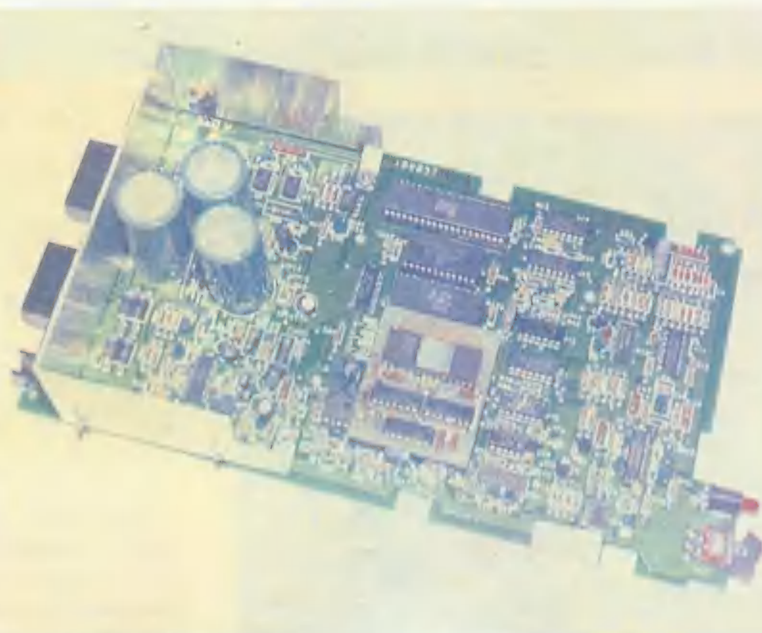
taw kilku bardzo użytecznych i wbudowanych na stałe w pamięć ROM programów narzędziowych, przeznaczonych głównie do celów diagnostycznych oraz do kopiowania dysków, „łamania" zabezpieczeń przed kopiowaniem i tworzenia własnych zabezpieczeń.

Pierwszy z wymienionych programów, o nazwie TOP COPY, służy głównie do kopiowania dyskietek, w tym także zabezpieczonych. TOP COPY ma kopiować praktycznie wszystkie programy, które zapisano bez specjalnych środków technicznych w rodzaju laserowych znaczników lub specjalnych formaterów. Wynika to z faktu, że TOP COPY dokładnie analizuje format zapisu na dyskietce oryginalnej, w pełni wykorzystując informacje, dostarczane przez kontroler. Zaletą TOP COPY jest możliwość wykorzystania rozszerzonej pamięci operacyjnej, o ile ta tylko jest obecna (np. 128 lub 256 KB). Pozwala to na kopiowanie dyskietek bez ich uciążliwego przekładania.

Następny program narzędziowy — TOP FORMATTER 720 — jest przeznaczony do analizy sposobu formatowania dyskietek oraz do tworzenia własnych formatów dysku przy wykorzystaniu wszystkich możliwości oferowanych przez zainstalowany w stacji 1050 kontroler. Analiza formatu przydaje się zazwyczaj przy „łamaniu" zabezpieczonych programów, natomiast specjalne, samodzielnie definiowane formaty pozwalają zabezpieczać własne programy. TOP FORMATER pozwala formatować oddzielnie każdą z 40 ścieżek. Interesujący jest zwłaszcza fakt, że FORMATER pozwala sformatować ścieżkę na nowo bez utraty jej zawartości. Zawartość ścieżki jest ładowana do pamięci, ścieżka jest formatowana na nowo, po czym pierwotna zawartość jest wpisywana na dyskietkę ponownie, tym razem w nowym formacie. Możliwe jest m.in. tworzenie sektorów pustych, sektorów podwójnych (dwa lub więcej sektorów na ścieżce nosi identyczne oznaczenie) i sektorów z błędną sumą kontrolną. Można też oczywiście wpływać na liczbę sektorów na ścieżce.

Ostatni z programów narzędziowych — BLOK DIAGNOSTYCZNY — realizuje 2 zadania: testuje sprawność urządzenia oraz pozwala mierzyć i modyfikować prędkość obrotową dyskietki. Okazuje się, że stacja 1050 pozwala w pewnym stopniu zmieniać prędkość wirowania dyskietki w sposób programowy. Zmniejszając prędkość wirowania o ok. 7% można tworzyć nowe, interesujące systemy zabezpieczenia przed kopiowaniem lub uruchomieniem programu bez upoważnienia.

Zastosowanie TOP DRIVE nie powoduje jakichkolwiek ograniczeń standardowych funkcji stacji. Oznacza to, że zmodernizowana stacja wczytuje także bez problemów wszystkie programy, przeznaczone dla stacji oryginalnej. Zastosowany w TOP DRIVE format dyskietek funkcjonuje niezależnie od formatów już istniejących. Zastosowanie formatów standardowych nie pozwala jednak w pełni wykorzystać walorów urządzenia. Bardzo istotny jest fakt, że TOP DRIVE nie wymaga żadnej adaptacji dyskowego systemu operacyjnego (trzeba jednak zaznaczyć, że w systemie DOS 2.5 wykorzystanie podwójnej gęstości zapisu nie jest



możliwe). Wszystkie elementy oprogramowania, uczestniczące w przyspieszonej transmisji danych, są kopiowane do pamięci komputera wprost z zawartej w stacji pamięci EPROM. Do uruchomienia i eksploatacji TOP DRIVE nie jest zatem potrzebna żadna dodatkowa dyskietka z oprogramowaniem.

Nie wspomnieliśmy dotychczas, że poza usprawnieniem stacji dysków TOP DRIVE zawiera interfejs równoległy do drukarek, zgodny ze standardem Centronics (oprogramowanie interfejsu także jest zawarte w pamięci EPROM). Płytka TOP DRIVE zawiera złącze, do którego można dołączyć przewód zakończony zwyczajnym gniazdkiem diodowym, montowanym w obudowie stacji dysków. Do tego gniazdka można przyłączyć przewód drukarki, wyposażony na drugim końcu w standardowe, 36-stykowe drukarkowe złącze Centronics. Interfejs drukarki w TOP DRIVE jest w pełni zgodny z rozpowszechnionym, firmowym i niewspółmiernie drogim interfejsem typu ATARI 850. Jak nas informowano, istnieje także nieco tańsza wersja TOP DRIVE bez interfejsu Centronics.

TOP DRIVE jest przykładem, jak dzięki sprytnym sztuczkom programistycznym i trafnym kompromisom można istotnie ulepszyć sprzęt bez jego poważnych przeróbek. Poza układem EPROM, na płytce TOP DRIVE znajduje się tylko kilka układów, uczestniczących w transmisji danych pomiędzy stacją dysków, a komputerem oraz obsługujących interfejs Centronics.

Beczkę miodu może popsuć łyżka dziegciu. Smak dziegciu jest niestety wyczuwalny także w TOP DRIVE, chociaż na szczęście nie dotyczy to strony technicznej. Po pierwsze, razi angielska konwersacja i napisy. Skoro TOP DRIVE jest przeznaczony dla polskich użytkowników, to może mógłby komunikować się z nimi w języku Mikołaja Reja? Co gorsza, niektóre fragmenty dialogu są po polsku, inne po angielsku, co wskazywałoby na to, że firmę ATASERW trapią rozterki pomiędzy chęcią zachowania anglelskojęzycznego „fasonu" i usankcjonowania tubylczego dialektu. Lepsza mogłaby być także dokumentacja. Co prawda dostarczany z urządzeniem opis użytkowy liczy 10 stron, tym niemniej niektóre użyte w nim terminy i zjawiska nie zostały dostatecznie objaśnione. Przydałoby się więcej praktycznych przykładów. Poza tym lepsze mogłoby być językowe opracowanie instrukcji. Co prawda kiepski poziom instrukcji jest uniwersalną bolączką polskich wytwórców osprzętu i oprogramowania i na tym tle ATASERW wypada nawet całkiem nieźle, tym niemniej instrukcjom do naprawdę dobrych wyrobów należałoby postawić wyższe wymagania*[)].

TOP DRIVE może współpracować z każdym komputerem ATARI XL. W praktycznej eksploatacji z komputerem ATARI 800XL z pamięcią rozbudowaną do 256 KB TOP DRIVE sprawiał wrażenie bardzo korzystne i nie powodował żadnych kłopotów. Nie znaleźliśmy żadnego programu, którego nie dałoby się załadować z dyskietki i uruchomić pod kontrolą TOP DRIVE (ze skruchą przyznajemy, że do testowania wykorzystaliśmy głównie popularne gry). Szczególnie odczuwalna była podwyższona szybkość ładowania programów. Także współpraca z drukarką nie nasuwała żadnych uwag (do testów wykorzystaliśmy drukarkę STAR NL-10 z modułem typu EPSON).

Reasumując: naszym zdaniem TOP DRIVE to trafne i zręczne rozwiązanie technologiczne, które stosunkowo niewielkim kosztem i przy minimalnej ingerencji w sprzęt bardzo istotnie podnosi użyteczność komputerów klasy ATARI 800XL współpracujących ze stacją dysków typu 1050, zwłaszcza gdy idzie o zastosowania praktyczne.

*[)] Firma ATASERW obiecała uwzględnić nasze zarzuty w kolejnych edycjach swoich opracowań, które, jak nas zapewniono, są stale doskonalone.

GRZEGORZ ZALOT

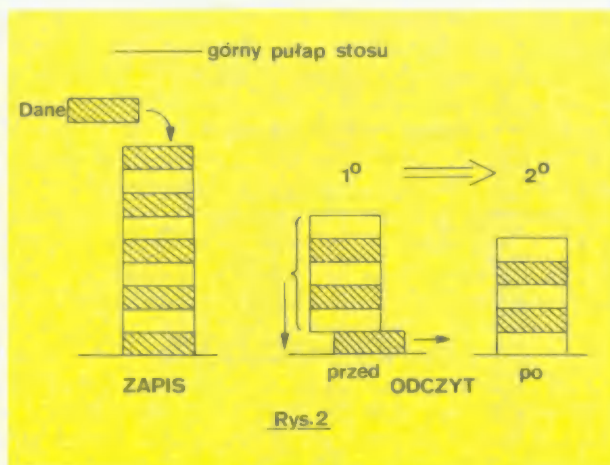
# PRZYSTOSOWANIE KLAWIATURY DO WSPÓŁPRACY Z EDYTOREM TASWORD

W pierwszym numerze naszego magazynu komputerowego opisaliśmy konstrukcję nieomal profesjonalnej klawiatury poprawiającej znacznie komfort pracy na popularnym Spectrum. Układ klawiatury umożliwia automatyczne symulowanie naciśnięcia klawisza normalnej klawiatury poprzedzonego wygenerowaniem trybu EXTENDED — a zatem nie jest konieczne dość kłopotliwe sekwencyjne naciskanie kilku klawiszy. Jak wiadomo najbardziej chyba popularny na Spectrum edytor tekstu Tasword w wersji polskiej generuje polskie litery poprzez kolejne włączenie trybu EXTENDED i naciśnięcie odpowiedniego klawisza. Wykorzystując więc klawiaturę w proponowanej wersji można uniknąć kłopotliwych sekwencji manipulowania klawiszami czyniąc pracę z edytorem znacznie bardziej efektywną. Dodatkowo możliwe jest zrealizowanie układu klawiszy praktycznie identycznego ze standardową maszyną do pisania — jest to niezmiernie ważne dla osób, które już wcześniej posiadły umiejętność maszynopisania. Drobne różnice dotyczą jedynie znaków niealfanumerycznych, umieszczonych nad górnymi klawiszami cyfrowymi; nieco inne znaczenie mają też klawisze na prawo od głównego pola. Klawisze funkcyjne oraz bloku numerycznego wykorzystano do bezpośredniego sterowania funkcjami edytora.

Znaczenie poszczególnych klawiszy przedstawia rys. 1. Układ podstawowej części nie zmienił się (kolor

← ↓ ↑ →
CAPS LOCK GRAPH
! 1 @ 2 # 3 $ 4 % 5 ↑ 6 & 7 * 8 ( 9 ) 0 + = DEL
EXT Q W E R T Z U I O P ENTER
SH Lock A S D F G H J K L
SHIFT Y X C V B N M , . SHIFT S.SW
SHIFT SPACE
ENT C.SW

Rys.1 Funkcje klawiszy w trybie Tasword (kolor czerwony oznacza funkcję inną, niż w trybie normalnym).



Rys. 2

czarny), dodano tylko w odpowiednich miejscach polskie litery (zmiany oznaczono kolorem czerwonym). Znaki, które były dostępne pod tymi klawiszami, zostały przeniesione na sąsiednie, względnie w przypadku rzadziej używanych, na klawisze pola numerycznego, będącego teraz właściwie polem funkcyjnym (analogiczne rozwiązania stosuje się w klawiaturach profesjonalnych).

Klawisze funkcyjne F1 do F10 mają w trybie Tasword następujące znaczenie (od lewej):

— wstawianie znaku tekstu
— zsuń akapit
— kursor na początek tekstu
— kursor na koniec tekstu
— przesuw tekstu o 1 linię w dół
— przesuw tekstu o 1 linię w górę
— znak '~', który nie zmieścił się na innych klawiszach
— przesuw tekstu do lewego marginesu
— przesuw tekstu do prawego marginesu
— centrowanie tekstu.

Klawisze sterowania ruchem kursora mają normalne znaczenie z tym wyjątkiem, że przy naciśniętym SHIFT (dowolnym, gdyż w tym trybie oba SHIFT-y mają takie samo znaczenie) kursor przeskakuje przez wyraz.

W prawym bloku funkcyjnym od lewej klawisz CAPS LOCK nie zmienił znaczenia, natomiast następny kasuje linię, jednak tylko z włączonym SHIFT. Dalej GRAPH i powrót do BASIC-a (czyli STOP), także tylko z SHIFT.

Znaczenie klawiszy funkcyjnych z pierwszych dwóch rzędów bloku numerycznego jest następujące (po znaku ^ znaczenie z SHIFT):

górny rząd od lewej:
— tryb rozsuwania wł/wył, ^ początek bloku
— tryb przenoszenia wł/wył, ^ koniec bloku
— tryb wstawiania wł/wył, ^ kopiuj blok
— tryb 32×64 znaki, ^dosuń blok

drugi od góry rząd od lewej:
— ust. lewego marginesu, ^ zsuń linię
— ust. prawego marginesu, ^ rozsuń linię
— kasowanie marginesów, ^ szybkie przewijanie tekstu do tyłu
— szukaj wyraz, ^ szybkie przewijanie tekstu do przodu.

Znaczenie pozostałych klawiszy pokazano na rys. 1.

Jedyną funkcją, która nie znalazła się bezpośrednio na klawiszach, jest kasowanie bufora tekstu — realizujemy je poprzez znane EXT x. Funkcja HELP realizowana jest przez EDIT.

```
;POPRAWKI TASWORDA

;POPR. PRZEN. POLSKICH LITER
                ORG  #FD08
                EX   DE,HL
                LD   HL,#F300
                BIT  3,(HL);TESTUJ WSK.
                JP   #E720

                ORG  #E720
                JR   Z,DALEJ
                CP   #0E;TESTUJ EXT MODE
                JR   Z,DALEJ
                JP   #FFDF
DALEJ           EX   DE,HL
;ODTAD BEZ ZMIAN

;WYL TRYBU EXT DLA ZNAKOW SPECJ.
                ORG  #F4EA
                JP   EXTOFF

;WEKTOR PRZERWANIA
                ORG  #E5FF
                DEFW INTIM2

                ORG  #ECBD
                JP   WYLP2;WYJSCIE DO BASIC

;ZWOLNIENIE MIEJSCA
;NA WEKTOR PRZERWANIA
                ORG  #E5FC
                JP   PROGR

;ZMIANA WEJSCIA DO PROGRAMU
                ORG  #FB4A
                CALL #FA7A
                CALL ZERBUF
                CALL #FA5E
POB             CALL #FAC8
                CALL POBZN;POBIERZ ZNAK
                CP   #00;ZERO DLA BRAKU
                JR   Z,POB
                JP   #F321

;POPRAWKI TABLICY SKOKOW
;DLA FUNKCJI ZX PRINTER
                ORG  #F500
                DEFW #F7F5
                ORG  #F50C
                DEFW #F7F5
                ORG  #F518
                DEFW #F7F5
                ORG  #F53A
                DEFB #6E;POPR. DOSUN BLOK

;USUNIECIE BEEP
                ORG  #F325
                NOP
                NOP
                NOP

;ZMIANA BORDER NA 6
                ORG  #FC04
                DEFB #06

;PROGRAM

                ORG  #F347
;WYL. TRYBU 2 PRZERWAN
;DLA WYJSCIA DO BASIC
                IM   1
                RET

;PROCEDURA OBSLUGI KLAWIATURY

INTIM2          PUSH AF;ZAPAM. REJESTRY
                PUSH BC
                PUSH DE
                PUSH HL
                LD   A,(#5C08)
                CP   0;CZY NOWY ZNAK
                JR   Z,WYJ
JESTZN          PUSH AF;JEST NOWY ZNAK
                CALL #E711;BEEP
                LD   HL,(WSKA)
                LD   A,L
                INC  A
                AND  #0F
                CP   H;CZY MIEJSCE W BUFORZE
                JR   NZ,WCZYT
                POP  AF;NIE MA MIEJSCA
                JR   WYJ

WCZYT           SUB  A;JEST MIEJSCE
                LD   (#5C08),A
                LD   HL,BUFOR
                LD   B,0
                LD   A,(WSKA)
                LD   C,A
                ADD  HL,BC;OBL.ADRES
                POP  AF;ODTWORZ ZNAK
                LD   (HL),A;ZAPISZ
                INC  C;WSKAZN.+1
                LD   A,C
                AND  #0F
                LD   (WSKA),A
WYJ             POP  HL;ODTWORZ REJESTRY
```

```
1080               POP  DE
1090               POP  BC
1100               POP  AF
1110               JP   #0038;SKOK DO ROM
1120
1130 WSKA          DEFB 0
1140 WSKB          DEFB 0
1150
1160 BUFOR         DEFS #10;MIEJSCE NA BUFOR
1170
1180 ;PROCEDURA ODCZYTU ZNAKU
1190
1200 POBZN         LD   HL,(WSKA)
1210               LD   A,H
1220               CP   L;CZY NOWY ZNAK
1230               JR   NZ,NOWYZN;JEST
1240               SUB  A;NIE - A=0
1250               RET
1260
1270 NOWYZN        LD   HL,BUFOR
1280               LD   C,A
1290               LD   B,0
1300               ADD  HL,BC;ADR. W BUFORZE
1310               LD   B,(HL)
1320               INC  A;AKT. WSKAZNIK
1330               AND  #0F
1340               LD   (WSKB),A
1350               LD   A,B;ZNAK DO A
1360               RET
1370
1380 ZERBUF        PUSH HL;INICJACJA BUFORA
1390               LD   HL,0
1400               LD   (WSKA),HL
1410               LD   A,#E5
1420               LD   I,A;ST. BAJT WEKT. PRZERW
1430               POP  HL
1440               SUB  A
1450               LD   (#5C08),A
1460               IM   2;TRYB 2 PRZERWAN
1470               JP   #FA43
1480
1490
1500               ORG  #FD93
1510 EXTOFF        CALL #F5F7;WYL. TRYB EXT
1520               JP   #FC7B
1530
1540 PROGR         CALL #FA65;INNE WEJSCIE
1550               LD   (HL),#92
1560               JP   #E5B9
1570
1580 WYLP2         POP  HL;WYJSCIE DO BASIC
1590               POP  HL
1600               POP  HL
1610               IM   1
1620               RET
1630
1640 ;WYLACZENIE PRZERWAN PRZY HELP
1650               ORG  #F3E9
1660
1670 EDITEX        IM   1
1680               CALL #EC88
1690               JR   POWROT
1700
1710 EDITN         IM   1
1720               CALL #F5CF
1730               JR   POWROT
1740
1750 POWROT        SUB  A
1760               LD   (#5C08),A
1770               IM   2
1780               RET
1790
1800               ORG  #EAA5
1810
1820 ;WYL. TRYBU EXT DLA SZYBKIEGO SCROLL
1830 FASTSF        CALL #F5C1
1840               JR   WYEXT
1850
1860 FASTSG        CALL #F585
1870               JR   WYEXT
1880
1890 ZSUN          CALL #F717;TEZ ZSUN
1900               JR   WYEXT
1910
1920 ROZSU         CALL #FB7E;I ROZSUN LINIE
1930 WYEXT         JP   #F5F7
1940
1950 ;ROZPOZN. CAPS SH DLA
1960 ;POLSKICH LITER
1970
1980 ;ADRESY TABLIC KODOW
1990 TABKZR        EQU  #5F22
2000 TABKWY        EQU  #5E6C
2010 PRZEKO        EQU  #5E54
2020
2030 WEJP          LD   HL,TABKZR
2040               LD   DE,TABKWY
2050               CALL PRZEKO;TEST
2060               JP   NZ,#F4AC
2070               CP   #0F;POL. LITERA
2080               JR   C,DLIT
2090               LD   HL,#F300
2100               BIT  7,(HL);TEST CAPS LOCK
2110               JR   Z,DLIT
2120               LD   BC,#0009;POPRAWKA
2130               EX   DE,HL
2140               AND  A
2150               SBC  HL,BC;OBL. ADRES
2160               LD   A,(HL);LADUJ DUZA LIT.
2170 DLIT          CALL #F5F7;POWROT
2180               JP   #FF37
2190
2200 ;POPRAWA SKOKU DLA POL. LITER
2210               ORG  #FC79
2220               DEFW WEJP
2230
2240               ORG  #EA67
2250
2260 KASUJ         IM   1;KAS. BUF. TEKSTU
2270               CALL #F2EB
2280               LD   HL,#F300
2290               BIT  4,(HL)
2300               CALL NZ,#F5F7
2310               JP   POWROT
2320
2330 ;POPRAWA ADRESOW SKOKOW PROCEDUR
2340 ;STARSZY BAJT NA MLODSZEJ POZYCJI
2350
2360               ORG  #F4FA
2370               DEFW #A5EA
2380               ORG  #F4FD
2390               DEFW #AAEA
2400               ORG  #F503
2410               DEFW #AFEA
2420               ORG  #F506
2430               DEFW #B4EA
2440               ORG  #F509
2450               DEFW #E9F3
2460               ORG  #FCCA
2470               DEFW #F0F3
2480               ORG  #F512
2490               DEFW #67EA
2500
2510 ;*KONIEC
```

W jaki sposób wprowadzamy naszą klawiaturę w tryb umożliwiający współpracę z Taswordem? Spójrzmy na rys. 2c z numeru I 1987 „InforMika". Jest na nim zaznaczony przełącznik MODE, włączający normalnie nie wykorzystaną część pamięci EPROM programującej znaczenie klawiszy. Wystarczy zatem w tej pamięci doprogramować komórki o adresach powyżej 256D (200H), no i dodatkowo opisać niektóre klawisze. Dane do zaprogramowania pamięci (szesnastkowo) zawiera tablica 1.

### Poprawiamy Tasword

Nie jest to jednak koniec prac umożliwiających wygodne posługiwanie się edytorem. Przede wszystkim nasza klawiatura jest zbyt szybka, jak na możliwości Tasworda — następowałoby gubienie znaków, szczególnie podczas automatycznego przenoszenia wyrazów i rozsuwania linii. Ponadto niektóre funkcje (np. szybki scrolling) nie wyłączają trybu EXT, co w tym przypadku jest raczej niepożądane. Osobiście jestem zdania, że jeżeli już coś się robi, to raczej porządnie i do końca, a zatem wskazane byłoby poprawienie przy okazji kilku istotnych wad Tasworda, jakimi są np. nieprzenoszenie polskich liter do nowej linii, gdy są one pierwszym znakiem w tej linii, nierozpoznawanie CAPS LOCK dla polskich liter, nierealizowanie funkcji DOSUN BLOK, brak bufora wprowadzanych znaków. Zrealizowanie owych zmian wymaga dość głębokiego zaingerowania w program napisany w języku maszynowym i pełne wyjaśnienie zagadnienia przekracza nieco ramy tego artykułu. Spróbujemy jednak choć częściowo opisać zmiany dające znaczną poprawę jakości pracy edytora.

Wprowadzone zmiany nie zmieniają objętości programu, jednak wygospodarowania obszaru koniecznego dla wprowadzenia fragmentów programu dokonano drogą likwidacji funkcji przewidzianych dla ZX Printer (także Seikosha GP 50) — jest to drukarka używana stosunkowo rzadko, a zatem ograniczenie to w praktyce jest niemal zupełnie do pominięcia.

Przeanalizujmy program źródłowy generujący poprawki w głównym bloku Tasworda, liczącym 10751 bajtów — listing 1.

Pierwszą poprawką, nie wymagającą rezerwowania miejsca, jest realizacja przenoszenia polskich liter do nowej linii. Pokazują to linie listingu od nr. 40 do 150. Fragment programu od FDO8H jest realizowany każdorazowo po przyjęciu nowego znaku z klawiatury. Trzeci bit komórki F300H informuje, czy kursor stoi na początku nowej linii i czy zachodzi konieczność ewentualnego przeniesienia wyrazu. Gdy jest on wyzerowany, nie ma takiej sytuacji, natomiast gdy jest ustawiony (charakterystyczny duży kursor na początku nowej linii), testujemy znak z akumulatora na kod EXT (14D, czyli OEH). Gdy jest to EXT, nie zrealizujemy procedury przeniesienia (etykieta DALEJ), gdy jest to inny znak, skaczemy do innej części programu (FFDFH).

Linie 190 i 200 realizują wyłączenie trybu EXT dla wprowadzania znaków specjalnych, np. {, }, | itd., poprzez skok do procedury EXTÓFF, wyłączającej tryb EXT.

Realizacja bufora danych wprowadzanych z klawiatury wymaga pracy procesora w drugim trybie przerwań — wektor obsługi przerwania musi być umieszczony pod adresem zawierającym w młodszym bajcie same jedynki — w naszym przypadku E5FFH (linie 230—240). Konieczne jest jednak przeniesienie fragmentu programu z tego miejsca w inne. Pokazano to w liniach 310—320.

Kolejne linie 350—430 zmieniają nieco pierwsze wejście do Taswordа. Oprócz występujących normalnie wywołań podprogramów dochodzi podprogram zerowania bufora klawiatury ZERBUF, zmienia się także procedura odczytu znaku, w tym przypadku z bufora — POBZN. Dla zera w akumulatorze, co oznacza brak nowego znaku, realizowana jest pętla oczekiwania (POB).

Linie od 470 do 540 realizują zmianę adresów procedur ZX Printer: skok do F7F5 jest powrotem bez wykonywania jakichkolwiek czynności. W ten sposób uzyskujemy wolne miejsce do umieszczenia własnych procedur. Przy okazji zmieniamy w linii 540 błędną daną w adresie procedury DOSUN BLOK — błąd występuje w jednym bicie!

Linie od 570 do 600 usuwają BEEP (a w zasadzie tzw. klick) w głównej pętli rozpoznawania znaków Taswordа — jest on przeniesiony do programu obsługi bufora.

Od linii 660 rozpoczynają się właściwe procedury dopisane do Taswordа. Zaczyna się to w linii 710 wyłączeniem drugiego trybu przerwań przy jednym z wyjść do BASIC-a, konieczne dla poprawnej pracy interpretera. Odtąd procedury w listingu są nieco pomieszane — wynika to z faktu dość trudnego „upychania" fragmentów kodu w pustych miejscach po procedurach ZX Printer. Rozpoczynają się one programem obsługi przerwania, czyli obsługi bufora klawiatury (linie od 760 do 1110). Zanim jednak wyjaśnimy działanie tej części programu, kilka słów na temat sensu stosowania i rodzajów buforów klawiatury.

### Bufory klawiatury

Czym jest bufor? W dość dużym przybliżeniu jest to element pośredniczący między dwoma fragmentami pewnej całości, elastycznie je łączący. Porównanie do buforów wagonów kolejowych nie jest tutaj może najlepsze, ale i one w pewnym stopniu przecież niwelują skutki różnych prędkości. Lepiej jest porównać zadania bufora klawiatury do kolejki interesantów czy raczej

| Adres | Dane | Adres | Dane | Adres | Dane |
|---|---|---|---|---|---|
| 0200 | 75 | 0220 | 37 | 0240 | 7F |
| 0201 | 61 | 0221 | 02 | 0241 | 25 |
| 0202 | 65 | 0222 | 0A | 0242 | 10 |
| 0203 | 6D | 0223 | 12 | 0243 | 19 |
| 0204 | 71 | 0224 | 1A | 0244 | 21 |
| 0205 | 69 | 0225 | 24 | 0245 | 26 |
| 0206 | D1 | 0226 | 1D | 0246 | 17 |
| 0207 | 52 | 0227 | 15 | 0247 | 6F |
| 0208 | 5A | 0228 | 0D | 0248 | A3 |
| 0209 | 62 | 0229 | 8C | 0249 | 9C |
| 020A | 4B | 022A | 70 | 024A | 06 |
| 020B | 4C | 022B | 3C | 024B | 7F |
| 020C | 33 | 022C | 81 | 024C | BB |
| 020D | 7F | 022D | 91 | 024D | CC |
| 020E | 34 | 022E | 89 | 024E | BC |
| 020F | 7F | 022F | 9A | 024F | 06 |
| 0210 | 2B | 0230 | 7F | 0250 | 7F |
| 0211 | 03 | 0231 | 01 | 0251 | 7F |
| 0212 | 0B | 0232 | 09 | 0252 | 18 |
| 0213 | 13 | 0233 | 11 | 0253 | 20 |
| 0214 | 1B | 0234 | 22 | 0254 | 27 |
| 0215 | 23 | 0235 | 08 | 0255 | 1F |
| 0216 | 1C | 0236 | 1E | 0256 | 07 |
| 0217 | 14 | 0237 | 16 | 0257 | 67 |
| 0218 | 0C | 0238 | 0E | 0258 | 6E |
| 0219 | 04 | 0239 | 05 | 0259 | 8B |
| 021A | 5E | 023A | 6C | 025A | 5D |
| 021B | 44 | 023B | 2C | 025B | 7F |
| 021C | 92 | 023C | B3 | 025C | 6A |
| 021D | 8A | 023D | C3 | 025D | 72 |
| 021E | 95 | 023E | C4 | 025E | 7F |
| 021F | 98 | 023F | AC | 025F | 7F |
| 0280 | 75 | 02A0 | 37 | 02C0 | 7F |
| 0281 | 61 | 02A1 | 2A | 02C1 | 4D |
| 0282 | 65 | 02A2 | 32 | 02C2 | 38 |
| 0283 | 6D | 02A3 | 3A | 02C3 | 41 |
| 0284 | 71 | 02A4 | 42 | 02C4 | 49 |
| 0285 | 69 | 02A5 | 76 | 02C5 | 4E |
| 0286 | D1 | 02A6 | 45 | 02C6 | 3F |
| 0287 | 52 | 02A7 | 3D | 02C7 | 94 |
| 0288 | 5A | 02A8 | 35 | 02C8 | CB |
| 0289 | 62 | 02A9 | B4 | 02C9 | 84 |
| 028A | 3B | 02AA | 55 | 02CA | 06 |
| 028B | 4C | 02AB | 43 | 02CB | 7F |
| 028C | 33 | 02AC | A6 | 02CC | F5 |
| 028D | 59 | 02AD | 9E | 02CD | ED |
| 028E | 34 | 02AE | A1 | 02CE | E1 |
| 028F | 51 | 02AF | 99 | 02CF | 06 |
| 0290 | 2B | 02B0 | 7F | 02D0 | 7F |
| 0291 | 53 | 02B1 | 29 | 02D1 | 7F |
| 0292 | 5B | 02B2 | 31 | 02D2 | 40 |
| 0293 | 63 | 02B3 | 39 | 02D3 | 48 |
| 0294 | 6B | 02B4 | 4A | 02D4 | 4F |
| 0295 | 73 | 02B5 | 30 | 02D5 | 47 |
| 0296 | 74 | 02B6 | 46 | 02D6 | 07 |
| 0297 | 77 | 02B7 | 3E | 02D7 | A4 |
| 0298 | 64 | 02B8 | 36 | 02D8 | 93 |
| 0299 | 5C | 02B9 | 2D | 02D9 | 9B |
| 029A | 66 | 02BA | 68 | 02DA | 58 |
| 029B | 44 | 02BB | 2C | 02DB | 7F |
| 029C | A7 | 02BC | E9 | 02DC | D5 |
| 029D | A0 | 02BD | F1 | 02DD | 60 |
| 029E | 9F | 02BE | D9 | 02DE | 7F |
| 029F | 97 | 02BF | 54 | 02DF | 7F |
| 0300 | 75 | 0320 | 37 | 0340 | 7F |
| 0301 | 61 | 0321 | 2A | 0341 | 4D |
| 0302 | 65 | 0322 | 32 | 0342 | 38 |
| 0303 | 6D | 0323 | 3A | 0343 | 41 |
| 0304 | 71 | 0324 | 42 | 0344 | 49 |
| 0305 | 69 | 0325 | 76 | 0345 | 4E |
| 0306 | D1 | 0326 | 45 | 0346 | 3F |
| 0307 | 52 | 0327 | 3D | 0347 | 94 |
| 0308 | 5A | 0328 | 35 | 0348 | CB |
| 0309 | 62 | 0329 | B4 | 0349 | 84 |
| 030A | 3B | 032A | 55 | 034A | 06 |
| 030B | 4C | 032B | 43 | 034B | 7F |
| 030C | 33 | 032C | A6 | 034C | F5 |
| 030D | 59 | 032D | 9E | 034D | ED |
| 030E | 34 | 032E | A1 | 034E | E1 |
| 030F | 51 | 032F | 99 | 034F | 06 |
| 0310 | 2B | 0330 | 7F | 0350 | 7F |
| 0311 | 53 | 0331 | 29 | 0351 | 7F |
| 0312 | 5B | 0332 | 31 | 0352 | 40 |
| 0313 | 63 | 0333 | 39 | 0353 | 48 |
| 0314 | 6B | 0334 | 4A | 0354 | 4F |
| 0315 | 73 | 0335 | 30 | 0355 | 47 |
| 0316 | 74 | 0336 | 46 | 0356 | 07 |
| 0317 | 77 | 0337 | 3E | 0357 | A4 |
| 0318 | 64 | 0338 | 36 | 0358 | 93 |
| 0319 | 5C | 0339 | 2D | 0359 | 9B |
| 031A | 66 | 033A | 68 | 035A | 58 |
| 031B | 44 | 033B | 2C | 035B | 7F |
| 031C | A7 | 033C | E9 | 035C | D5 |
| 031D | A0 | 033D | F1 | 035D | 60 |
| 031E | 9F | 033E | D9 | 035E | 7F |
| 031F | 97 | 033F | 54 | 035F | 7F |

kolejki papierów na biurku biurokraty. Jeżeli na biurku tym zmieściłby się tylko jeden dokument, to przed końcem jego opracowania inne napływające papiery byłyby po prostu zagubione. Tak samo dzieje się w edytorze: czasem klawiatura dostarcza znaki szybciej niż zdąży przetworzyć je edytor. Aby nie następowało gubienie informacji, należy wprowadzić „kolejkę", najlepiej zgodnie ze znaną zasadą: „Kto był pierwszy, będzie pierwszy załatwiony i dalej według kolejności". Nie ma tu uprzywilejowanych i rzeczywiście kolejność jest zachowana. Biurokrata czyni to kładąc nowe papiery na górę stosu dokumentów i biorąc nowe ze spodu (względnie odwrotnie — ważny jest skutek). Stos taki w informatyce nazywa się stosem FIFO (First In, First Out), w odróżnieniu od typowego stosu maszynowego procesora, będącego stosem LIFO (Last In, First Out).

Ilustracją działania stosu FIFO jest rys. 2. Zapis, to dodanie jednego elementu na górę. I tu mała UWAGA! — należy kontrolować poziom stosu, aby nie doprowadzić do jego przepełnienia. Odczyt, to wyciągnięcie elementu z dołu i przesunięcie pozostałych elementów o jedno „piętro" niżej. Jest to intuicyjnie najprostszy rodzaj stosu, jednak przy jego realizacji w asemblerze wykonujemy niepotrzebnie kilka czynności, np. przesuwanie zawartości w dół. Wynika to z faktu, że poziom najniższy możemy sobie ustalić zupełnie względnie, może się on np. zmieniać przy odczycie (podnosić). Aby jednak nie marnować miejsca, stos taki możemy zapętlić, tworząc kołową kolejkę z wydzielonym początkiem i końcem, obracającą się jakby wokoło, lecz bez poruszania elementów. Daje to już znaczną oszczędność operacji i czasu. Spójrzmy na rys. 3a. Wskaźnik odczytu reprezentuje najniższy poziom stosu, natomiast wskaźnik zapisu — jego górę, konkretnie pierwsze wolne miejsce. A zatem zapisujemy według wskaźnika zapisu, po czym zwiększamy jego wartość o jeden. Oczywiście w tym przypadku ze względu na kołowy (powtarzalny) charakter zjawiska arytmetykę prowadzimy modulo pojemność stosu (patrz rysunek 16). Odczyt polega na pobraniu elementu według wskaźnika odczytu i następnej inkrementacji tegoż wskaźnika. Kiedy stos jest pusty? Następuje to wtedy, gdy wskaźnik zapisu (pierwsze wolne miejsce) jest równy wskaźnikowi odczytu. Pełne zapełnienie następuje natomiast wtedy, gdy po ewentualnym zapisie wskaźnik zapisu powiększony o jeden zrównałby się ze wskaźnikiem odczytu (rys. 3b). W ten sposób jakby tracimy jedno miejsce, lecz jest to element odróżniający stan pełnego zapełnienia od pustego stosu. Można to oczywiście wykonać inaczej, ale kosztem nieopłacalnej komplikacji algorytmu.

### Analizujemy program

Przejdźmy teraz do analizy programu. Na początek zapis danych do bufora (linie od 760). Ten fragment jest wywoływany cyklicznie przez przerwanie 50 razy na sekundę — a zatem należy najpierw zapamiętać na stosie maszynowym wszystkie używane rejestry. Następnie odczytujemy komórkę 5C08H — LAST K. Gdy jej zawartość jest różna od zera, standardowa procedura ROM odczytała nowy znak z klawiatury. Gdy nie, skaczemy do WYJ i po odtworzeniu rejestrów wykonujemy procedurę obsługi przerwania z ROM (linia 1110).

Gdy mamy nowy znak (JESTZN), zapamiętujemy akumulator (z kodem znaku) na stosie, generujemy BEEP, a następnie testujemy, czy jest miejsce w buforze (stosie FIFO). W tym celu pobieramy do L (WSKA) (wskaźnik zapisu), do H — (WSKB) (wskaźnik odczytu — ładowane razem, jako para), inkrementujemy L przepisane do A (linia 860,870), oczywiście modulo 10H (linia 880). Gdy nie ma miejsca, wynik zerowy, odczyt A ze stosu (linia 910) i dalej do wyjścia (WYJ). Ponieważ w tym miejscu nie zmieniamy zawartości LAST K, program generuje charakterystyczne brzęczenie o częstotliwości 50 Hz — informuje ono o przepełnieniu bufora. Praktycznie występuje to jedynie dla wpisu znaków z funkcją REPEAT.

Gdy jest wolne miejsce w buforze, wpisujemy do LAST K zero (informacja, że znak został odczytany) (940,950), do HL adres początkowy bufora, do B zero, do C — wskaźnik zapisu, przechowywany pod adresem WSKA, poprzez akumulator. Poprawkę z BC dodajemy do HL w arytmetyce 16-bitowej i po odtworzeniu kodu znaku (1010) zapisujemy do bufora daną (1020). Pozostaje jeszcze inkrementacja wskaźnika modulo 10H (1030—1050) i zapis pod adresem WSKA. Dalej, jak bez nowego znaku.

Linie 1130 do 1160 rezerwują miejsce na wskaźniki zapisu i odczytu oraz na bufor (16 bajtów).

Od linii 1200 rozpoczyna się procedura odczytu znaku z bufora. Ładujemy oba wskaźniki do pary HL i porównujemy — gdy są sobie równe, bufor jest pusty i do akumulatora wpisujemy zero (linia 1240) informujące o braku nowych znaków. Gdy wskaźniki nie są sobie równe (NOWYZN), oznacza to znak do odczytu. Do HL ładujemy adres początku bufora, do BC poprawkę adresu i po obliczeniu adresu elementu do odczytu (linia 1300) odczytujemy znak — ładujemy do rejestru B. Jeszcze inkrementacja wskaźnika odczytu (linia 1320), modulo 16 (operacja AND 0FH) i jego zapis pod adres WSKB. Operację odczytu znaku kończy przepisanie jego kodu do akumulatora (linia 1350).

Następna procedura, to wstępne wyzerowanie bufora i zainicjowanie jego pracy. Na początku zapamiętujemy na stosie parę HL (linia 1380), wykorzystywaną w innych procedurach. Następnie zerujemy wskaźnik WSKA i WSKB (linia 1400), ładujemy E5H do rejestru I (linia 1420) tworzącego starszy bajt adresu, skąd odczytujemy wektor przerwania. Następnie odtwarzamy HL, zerujemy komórkę LASTK (linia 1440, 1450) — aby nie nastąpiło przypadkowe odczytanie jakiegoś znaku — oraz włączamy tryb 2 przerwań inicjując tym samym pracę bufora. W linii 1470 skaczemy do standardowej procedury inicjacji pracy edytora.

Procedura EXTOFF wyłącza tryb EXTENDED — jest ona wykorzystywana w celu wyłączenia tego trybu dla niektórych funkcji oraz znaków niealfanumerycznych. Fragment od etykiety PROGR jest ciągiem instrukcji, które wyrzucono przy rezerwacji miejsca na wektor przerwania. Procedura WYLP2 realizuje wyjście do BASIC-a przez zlecenie STOP — następuje „ściągnięcie" trzech adresów powrotów z podprogramów ze stosu, włączenie trybu 1 przerwań (konieczne dla poprawnej pracy interpretera BASIC — dla niewłączenia tego trybu może nastąpić zablokowanie komputera na skutek przepełnienia bufora).

Tryb 2 przerwań, czyli pracę bufora, należy także obowiązkowo wyłączyć przy funkcji HELP (linie od 1650). Wprowadzono dwie procedury, gdyż wywołanie

HELP może nastąpić w trybie normalnym oraz w trybie EXTENDED. Przy powrocie należy koniecznie wyzerować LAST K (linia 1760) oraz włączyć pracę bufora (czyli IM 2).

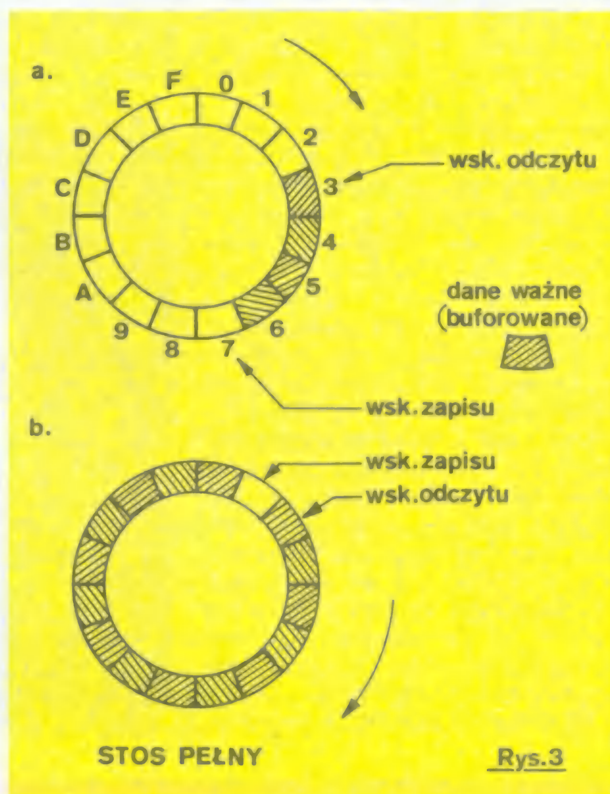
Następne dwie procedury wyłączają tryb EXT dla szybkiego rolowania ekranu o 22 wiersze (FASTSF i FASTSF) oraz zsuwania i rozsuwania linii — procedury te realizują najpierw skok do odpowiedniej procedury Taswordа (rozkaz CALL adr), a następnie skok do procedury wyłączającej tryb EXT (F5F7H). Skok do tychże miejsc programu następuje dzięki zmianom w tablicy adresów procedur, zrealizowanych w dalszej części programu.

Następny fragment programu realizuje funkcję rozpoznawania CAPS SHIFT dla polskich liter (od linii 2030). W tym fragmencie z konieczności wykorzystujemy częściowo blok programu dołożony do podstawowego bądź w linii O REM (najczęściej) lub też w postaci osobnego bloku bajtów (nosi on na ogół nazwę POL i ładuje się od adresu 31091 — ta wersja Taswordа przystosowana jest w zasadzie wyłącznie do współpracy z drukarką GF-500, za pomocą interfejsu Interface 1). Adresy procedur (linie 1990—2010) ustalone są dla wersji z O REM — dla innej wersji będą one inne i należy dokonać odpowiedniej ich zmiany.

Wejście procedury (linia 2030) rozpoczyna się wywołaniem procedury PRZEKO, z uprzednim podaniem w HL i DE adresów odpowiednich tablic, zawierających kody polskich liter wprowadzane z klawiatury (TABKRZ) oraz kody wynikowe (RABKWY). Wejście do tej procedury następuje każdorazowo po przyjęciu nowego znaku w trybie EXTENDED — dla wykrycia kodu polskiej litery następuje ustawienie wskaźnika zera oraz wstawienie kodu litery do akumulatora. Gdy nie wykryto polskiej litery, skaczemy do dalszej części programu (linia 2060). W linii 2070 testujemy akumulator na wartość mniejszą od OFH — gdy relacja jest spełniona, w akumulatorze jest kod dużej polskiej litery (por. kody polskich liter w Taswordzie — artykuł „Pomiędzy Sinclairem a Commodorem", „InforMik" nr 2/1987 s. 20) i można skoczyć do wyjścia procedury (linia 2170, DLIT). Gdy w akumulatorze jest kod małej polskiej litery, należy przetestować wskaźnik CAPS SHIFT — 7 bit komórki F300H. Gdy jest on ustawiony, włączony jest CAPS SHIFT, sterowanie nie przechodzi do etykiety DLIT (linia 2110). Należy teraz dokonać przekodowania. Korzystamy z tego, że para DE zawiera adres kodu małej polskiej litery w tablicy kodów wynikowych. W tejże samej tablicy, w jej pierwszej połowie, umieszczone są kody dużych polskich liter — a zatem od DE odejmujemy połowę długości tablicy, czyli 9 — linie 2130 — 2150. Teraz wystarczy do akumulatora załadować kod odpowiedniej dużej polskiej litery (linia 2160).

Pozostaje jeszcze zmiana skoku do procedury rozpoznawania polskich liter — linie 2210—2220 — tak, aby nie była realizowana stara wersja, znajdująca się w dodatkowym bloku, nie rozpoznająca CAPS SHIFT.

Kolejna, ostatnia już procedura KASUJ, realizuje kasowanie bufora tekstu, połączone z wyłączeniem pracy bufora klawiatury (włączenie IM i w linii 2260). Jest to konieczne dla wyeliminowania możliwości odczytu przez edytor znaków sterujących potwierdzeniem rozkazu kasowania tekstu. Ponadto należy wyłączyć tryb EXTENDED,



Rys. 3

jednak tylko wtedy, gdy był on uprzednio włączony. Testowanie włączenia tego trybu odbywa się w liniach 2280—2290 i w razie konieczności wywoływany jest podprogramem F5F7H. Taka procedura jest konieczna z tego powodu, iż kasowanie bufora tekstu może być wywołane przez zlecenie EXTx, a także wywoływane jest ono przy pierwszym uruchomieniu edytora, bez włączania trybu EXTENDED. Właściwe kasowanie realizowane jest przez podprogram F2EBH. Na końcu następuje skok do etykiety POWROT, czyli skasowanie zawartości LAST K oraz uruchomienie bufora klawiatury.

Końcowy fragment programu zmienia odpowiednio adresy odpowiednich procedur w tablicy Taswordа. Kłopot w tym miejscu polega na tym, że Tasword przechowuje starszy bajt adresu na młodszej pozycji, a przy pomocy asemblera GENS 3 nie jest możliwe wpisanie adresów etykiet w zmienionej kolejności. Jedyną rozsądną metodą jest przeprowadzenie asemblacji i następne ręczne wpisanie adresów procedur z zamienionymi miejscami bajtem starszym i młodszym w tychże adresach. Tak właśnie powstał fragment od linii 2360.

**Wykorzystujemy program źródłowy**

Na zakończenie jeszcze kilka słów, jak wykorzystać program źródłowy przedstawiony na listingu. Najpierw należy w pamięci umieścić asembler GENS 3, najlepiej pod adresem 30 000 (w żadnym wypadku program źródłowy nie może wyjść ponad 54783!). Następnie ładujemy główny blok Taswordа (10751 bajtów od adresu 54784), uruchamiamy asembler i wczytujemy program źródłowy (według listingu). Teraz wystarczy tylko uruchomić asemblację — asembler sam umieści kod wynikowy wewnątrz głównego bloku Taswordа. Po tej operacji blok ten zapisujemy na taśmie, w miejsce starej wersji — edytor nadaje się już do użytku.

Tak przygotowany zestaw do edycji tekstów (czyli przerobiona klawiatura oraz poprawiony i zmieniony Tasword — pod wieloma względami znacznie lepszy od innych edytorów na nim wzorowanych) jest naprawdę zupełnie niezłym narzędziem do pisania nawet względnie dużych tekstów. Największym mankamentem jest niemożność wygodnego operowania na większych zbiorach (ograniczona pamięć), a także utrudnienie i bardzo wolne kopiowanie fragmentów wewnątrz tekstu. Tym niemniej przy sprawnych urządzeniach peryferyjnych zestaw oddaje duże usługi, wcale niemałe, jak na jego cenę. Do innych, na szczęście drobnych mankamentów, można zaliczyć sporadyczne wpisywanie dużych polskich liter w miejsce małych, szczególnie przy szybkim pisaniu. Przekłamania zdarzają się także przy szybkim wprowadzaniu znaków, które wymagają włączenia trybu EXTENDED — wynika to z ograniczonej szybkości testowania stanu klawiatury przez komputer (teoretycznie 50 razy na sekundę, a zatem maksymalnie 25 znaków z EXT na 1 sekundę, praktycznie ok. 2—3 razy mniej), a także z kilku nie usuniętych błędów w Taswordzie. Do najciekawszych należy powielanie znaków w niektórych bardzo specyficznych sytuacjach przy zsuwaniu akapitu. Wady te jednak praktycznie nie wprowadzają znaczących utrudnień w pracy na tym pseudoprofesjonalnym zestawie do edycji tekstów.

**Od redakcji:** *Po wydrukowaniu w nr. 1/87 „InforMika" artykułu na temat podłączenia klawiatury do ZX Spectrum, otrzymaliśmy sporo listów z zarzutami: dla kogo to drukujecie? Kto zdobędzie taką klawiaturę, którą kupić można jakoby tylko w Zachodniej Europie? (klawiatury te pojawiły się już kilka miesięcy temu na giełdach i w wielu prywatnych sklepach elektronicznych). Ile osób wykona tę bardzo trudną przeróbkę? etc. etc. Czujemy się więc w obowiązku wyjaśnić, że oba artykuły Grzegorza Zalota służyć mają za WZÓR możliwości przerabiania Spectrum, Atari czy Commodore „na komputer", użyteczny do prac, które codziennie wykonujemy. Autor szczegółowo opisuje swój tok myślenia i pracy: pozwala to na powtórzenie tego cyklu przy zupełnie innym komputerze, klawiaturze, edytorze. Można, oczywiście, po prostu powtórzyć konstrukcję czy też przeróbkę Autora. To najprostsze i najłatwiejsze. Chcemy jednak podsunąć możliwości rozbudowy wszystkim tym, którzy z różnych (najczęściej finansowych) względów nie mogą pozwolić sobie na zakup IBM XT lub AT, a chcieliby pracować na własnym komputerze, o podobnych (choć tylko w pewnym wycinku) możliwościach.*

*Wdzięczni będziemy za listy, sugestie, uwagi: czy jest sens i potrzeba publikacji podobnych artykułów? Czekamy na listy! (red)*

## CIEKAWE KSIĄŻKI

Polski przekład książki Theo Pavlidisa **„Grafika i przetwarzanie obrazów"**, WNT, Warszawa 1987, który ukazuje się w 5 lat po wydaniu angielskim na pewno spotka się z dużym zainteresowaniem. Dziedziny, którym jest poświęcona książka — grafika komputerowa, przetwarzanie i rozpoznawanie obrazów — intensywnie się rozwijają. Szybki rozwój techniki powoduje, że maszyny o dużej mocy obliczeniowej, przetworniki wizyjne i graficzne urządzenia wyjściowe stają się coraz bardziej dostępne i tanie. Ich zastosowania, coraz liczniejsze i różnorodniejsze, rodzą wiele interesujących problemów, które przyciągają uwagę inżynierów wielu specjalności, programistów, informatyków, matematyków. Obecnie na świecie ukazuje się kilka czasopism naukowych poświęconych grafice, przetwarzaniu i rozpoznawaniu obrazów, niemal co rok wydawane są także monografie z tego zakresu. Jedną z nich jest właśnie omawiana książka.

Wydaje się, że z tworzonych metod i koncepcji powstaje gałąź nauki (nazwana przez autora komputerowym przetwarzaniem informacji obrazowej). Z trzech dziedzin, które autor do niej zalicza najdłuższą historię ma rozpoznawanie obrazów. W dziedzinie tej w Polsce ukazało się już kilka pozycji np. książki Kulikowskiego „Cybernetyczne układy rozpoznające" i Ajzermana „Rozpoznawanie obrazów...". Natomiast przetwarzanie obrazów i grafika komputerowa, na których autor się w swej książce koncentruje, są stosunkowo nowe. Jeśli nie liczyć popularnej książki Angella „Wprowadzenie do grafiki komputerowej" to omawiana pozycja jest pierwszą, poświęconą tym dziedzinom, ukazującą się po polsku monografią.

Intencją autora było przedstawienie zagadnień, które stanowią pewną bazę dla przetwarzania obrazów i grafiki komputerowej, pozostaną przez to aktualne jeszcze przez dłuższy czas. Wykład jest ścisły i precyzyjny. Autor kładzie nacisk na zrozumienie przez czytelnika omawianych zagadnień, stworzenie dobrych podstaw matematycznych i informatycznych dla wykładanego materiału. Książka jest bardzo dobrą podstawą do dalszych studiów. Sygnalizuje wiele problemów szczegółowych, dla których podawane są odnośniki do bardzo obszernej literatury. Dla przedstawianych metod podane są algorytmy, które je realizują. Algorytmy te mogą być podstawą do tworzenia przez czytelnika własnych programów użytkowych.

Do zagadnień przetwarzania obrazów, obok takich algorytmów, jak kodowanie, filtracja, segmentacja, ścienianie (szkieletowanie), włączony został także rozdział o rzutowaniu. Obejmuje on bardzo interesujące problemy odtwarzania obrazu na podstawie jego rzutów. Przedstawione algorytmy rekonstrukcji obrazu na podstawie rzutów mają zastosowanie w tomografii komputerowej.

Dla wielu czytelników najbardziej interesujące będą chyba jednak algorytmy związane z grafiką komputerową. Można ją podzielić na dwie grupy. Do pierwszej zaliczają się metody aproksymacji i interpolacji krzywych i powierzchni. Przedstawione tu algorytmy można znaleźć też w wielu kursach analizy numerycznej, zostały one jednak wybrane i rozwinięte pod kątem konkretnego zastosowania — w grafice komputerowej. W zakresie tych algorytmów książka będzie cenną pomocą dla wszystkich użytkowników mikrokomputerów, którzy zajmują się wykonywaniem obliczeń numerycznych i stykają się z problemem graficznego przedstawienia wyników.

W drugiej grupie znajdują się algorytmy dwu- i trójwymiarowej animacji. Mają one wiele efektownych zastosowań — począwszy od gier komputerowych, poprzez tworzenie tzw. ujęć komputerowych w filmach, a skończywszy na złożonych pakietach symulacji układów dynamicznych używanych np. do szkolenia pilotów, operatorów itp. Książka zawiera wyczerpujący przegląd związanych z animacją problemów. Przedstawiono algorytmy rozwiązywania podstawowych zagadnień: wycinania części obrazu, które na skutek ruchu stają się niewidoczne, a także przesłaniania jednych elementów obrazu przez inne. Omówione zostały też zagadnienia, które uważa się za zaawansowane: zróżnicowane oświetlenie różnych powierzchni, rzucanie cieni jednych elementów obrazu na inne.

**Andrzej Polański**

**PRZYPOMINAMY!**

— o naszej akcji pt. „Co dolega twojemu Spectrum?"
— czekamy na listy z propozycjami tematycznymi
— a także na interesujące propozycje artykułowe

Nie zapominajcie też, że najpewniejszą formą uzyskiwania numerów „INFORMIKA" jest prenumerata! Nasz numer indeksu: **366013**.

Zyczymy naszym Czytelnikom dobrej pracy z komputerami i „InforMikiem".



Zestaw: stacja TIMEX FDD 3000 i mikrokomputer ZX Spectrum w akcji. Monitor można postawić na obudowie stacji

LUCJAN KNAPCZYK
TADEUSZ RZEPECKI

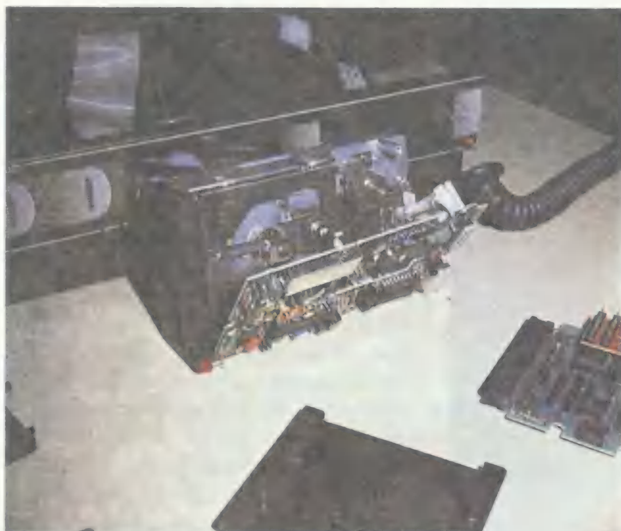
# STACJA DYSKÓW TIMEX FDD 3000

W ostatnim czasie pojawił się w sklepach Centralnej Składnicy Harcerskiej nowy wyrób — stacja dysków elastycznych FDD 3000 przeznaczona do współpracy z komputerami TIMEX 2048 lub ZX Spectrum. Urządzenie należy polecić wszystkim pragnącym poszerzyć swoje wiadomości na temat ZX Spectrum i wykorzystywać inne nośniki niż kasety z taśmą magnetyczną. Jest to dwukieszeniowa, trzycalowa stacja z wieloma oryginalnymi rozwiązaniami konstrukcyjnymi i programowymi. Na pierwszy rzut oka sprawia ona wrażenie solidnej i trwałej (a przy tym estetycznej) konstrukcji. Wykonana z grubej blachy obudowa umożliwia umieszczenie stacji pod monitorem, co radykalnie poprawia ergonomię pracy z zestawem. Do komputera urządzenie przyłączone jest spiralnym kablem poprzez interfejs. Dla użytkowników ZX Spectrum ważnym elementem jest przycisk RESET kasujący zawartość pamięci RAM części komputerowej zestawu.

Wnętrze stacji ujawnia całą istotę rozwiązania. Jest to praktycznie drugi komputer posiadający własny procesor Z80A i 16 kB pamięci ROM. Napędami steruje układ kontrolera WD 1770. Procesor jest taktowany zegarem szybszym niż w ZX Spectrum. Ponadto inne rozwiązania sprawiają, że czas wykonywania programów jest krótszy, niż w samym Spectrum. Sama płyta główna jest najprawdopodobniej zaadaptowana z innego systemu i być może posiada dodatkowe, interesujące możliwości wykorzystania. Wewnątrz obudowy i na płycie głównej istnieje kilka dosyć rażących prowizorek, charakterystycznych dla prototypu każdego urządzenia. Stacja wyposażona jest ponadto w gniazdo monitorowe, szkoda jednak, iż nie jest ono gniazdem typowym dla monitorów produkowanych w Polsce. Należy ponadto żałować, że stacja nie pozwala na zasilanie komputera kablem spiralnym poprzez interfejs i złącze krawędziowe komputera, cały zestaw byłby wtedy całkowicie pozbawiony plątaniny kabli.

Stacja dysków FDD 3000 umożliwia umieszczenie w kieszeniach dwóch napędów trzycalowych oznaczonych A i B. Dodatkowo istnieje możliwość dołączenia 2 napędów samodzielnych (sterownik umożliwia kontrolę 4 napędów), ale zasilanie tych dodatkowych musi być oddzielne. Dodatkowe napędy mogą być połączone poprzez łącza RS-232, których gniazda znajdują się z tyłu obudowy stacji. Umożliwiają one wykorzystanie komputera do pracy w sieci komputerowej, przesyłania danych do innych systemów oraz pracy z drukarką. Całość transmisji jest dokładnie opisana w instrukcji obsługi.

Praca komputera ze stacją możliwa jest pod kontrolą 2 systemów: stworzonego przez firmę TIMEX systemu TOS oraz najbardziej rozpowszechnionego systemu dla komputerów ośmiobitowych tzn. CP/M 2.2. System TOS działający pod kontrolą BASIC-a jest bardzo uniwersalny, a przy tym nie zajmuje pamięci RAM kom-

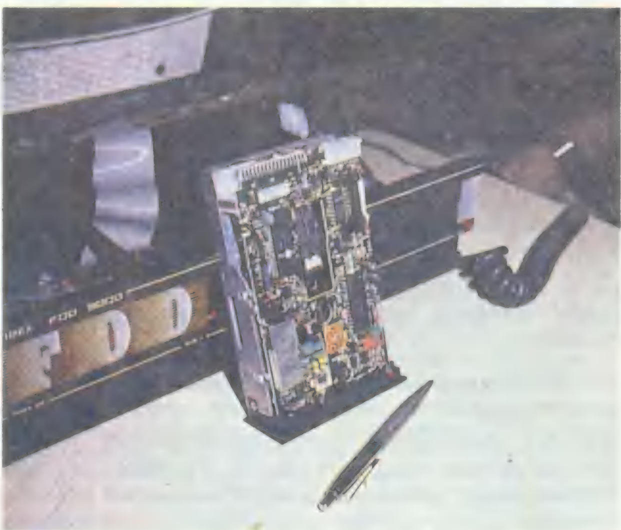


Trzycalowy napęd firmy Hitachi od środka — widoczna głowica i część mechaniczna

Wnętrze stacji FDD 3000 po zdjęciu obudowy



Napęd dyskowy charakteryzuje się znacznym stopniem „upakowania" elementów



putera. Umożliwia on posługiwanie się wszystkimi opcjami współpracy z pamięcią masową wraz z pewnymi rozszerzeniami (zainteresowanych można tutaj odesłać do książki K. Kuryłowicza, D. Madeja i J. Maraska pt. „Przewodnik po ZX Spectrum", gdzie działanie systemu TOS jest dość szczegółowo opisane). Ze strony praktycznej nasuwają się jednak pewne spostrzeżenia dotyczące obecnego oprogramowania komputerów ZX Spectrum, w szczególności gier. Większość programów wczytuje się z pominięciem instrukcji LOAD, przy wykorzystaniu odpowiedniej procedury umieszczonej w pamięci ROM komputera, wywoływanej z poziomu języka maszynowego. Ma to na celu zabezpieczenie przed nielegalnym kopiowaniem programów. Przeniesienie programu na dysk z poziomu języka maszynowego jest możliwe (odpowiednie procedury są opisane w instrukcji stacji), ale konieczna jest dość dobra znajomość programowania w języku maszynowym.

Przy pracy z systemem CP/M 2.2 komputer stanowi jedynie stację wprowadzania danych, funkcję podstawową przejmuje procesor znajdujący się w stacji. Jego zegar jest nieco szybszy niż w oryginalnym rozwiązaniu ZX Spectrum (4 MHz), ma on też do dyspozycji pełne 64 KB wolnej pamięci RAM oraz 16 KB ROM.

W czasie testów sprawdzono poprawność generacji systemu na nowej dyskietce dwoma sposobami:

**a.** za pomocą MOVCPM, a następnie SYSGEN. Przekopiowano w ten sposób sam system CP/M 2.2 na drugą dyskietkę;

**b.** przy pomocy dostarczonego programu kopiującego dyskietki.

Oba sposoby kopiowania dały sprawny system CP/M.

Szkoda, że nie dostarczono na dyskietce programu makroasemblera dla procesora Z80 oraz debuggera ZSTD, a jedynie programy ASM i DDT dostosowane do współpracy z rozkazami protoplasty Ziloga Z80 — mikroprocesora Intel 8080, mającego znacznie uboższą listę rozkazów (w instrukcji podaje się, że oba te programy są na dyskietce systemowej!). Jest to chyba jedyna krytyczna uwaga pod adresem systemu CP/M 2.2 używanego w stacji FDD 3000, poza tym system jest całkowicie sprawny i pozbawiony wady powracania po wykorzystaniu procedury systemowej ze zmienionym wskaźnikiem stosu, jak to ma miejsce w skopiowanych „produktach" niektórych firm polonijnych. Praca pod systemem CP/M 2.2 daje użytkownikowi możliwość korzystania z takich programów jak: MBASIC Microsoft, Pascal MT+ lub TURBO-Pascal, co znacznie rozszerza możliwości komputera.

Czas formatowania dyskietki w systemie TOS wynosi 31 sekund, natomiast w systemie CP/M 40 sekund. Przykładowy program testowy w języku maszynowym wykonywał się w systemie TOS 115 sekund, natomiast w systemie CP/M 101 sekund — różnica wynika z innej częstotliwości zegarów w obu przypadkach.

Stacja dysków TIMEX FDD 3000 sprawdziła się w praktyce w czasie eksploatacji zwiększając znacznie komfort pracy z komputerem. Trwałe i odporne na uszkodzenia mechaniczne dyskietki są bardzo wygodne w użyciu, chociaż stosunkowo drogie. Nieporozumieniem wydaje się być instrukcja opracowana w języku polskim, będąca niezbyt trafnie wybranym fragmentem instrukcji w języku angielskim.

DARIUSZ A. PRZYGODA

# DYSK PAMIĘCIOWY W ZX SPECTRUM (uzupełnienie)

Opisany w dwóch poprzednich numerach „InforMika" dysk pamięciowy spełnia swą rolę przy pracy z programami operującymi na strukturach kodu maszynowego (asembler, monitor) lub językami wyższego rzędu (Pascal, C). Okazuje się jednak, że często program obsługi dysku pamięciowego może odmówić prawidłowej współpracy z programami napisanymi w języku BASIC.

Przyczyną tego zjawiska jest cecha systemu operacyjnego mikrokomputera (system ten wraz z interpreterem języka BASIC umieszczony jest w pamięci stałej ROM), polegająca na traktowaniu pamięci ROM jako obszaru, który z definicji jest niezapisywalny. Założenie takie umożliwia niekiedy znaczne uproszczenie struktury programu, gdyż pozwala na traktowanie obszaru pamięci ROM jako swoistego „śmietnika".

W przypadku programu obsługi dysku pamięciowego, który umieszczony jest w pamięci o dostępie swobodnym RAM nakładkowanej na nie używany obszar pamięci ROM, pewne procedury edytora systemowego powodowały zamazywanie jego fragmentów, uniemożliwiając tym samym prawidłową pracę.

Zapobiec temu zjawisku można za pomocą opisanych poniżej prostych przeróbek sprzętowych i programowych, powodujących czasowe wzbronienie dostępności nakładkowanej pamięci do zapisu. Idea rozwiązania polega na blokowaniu sygnału /WE zapisu do nakładkowanej pamięci RAM. Sterowanie procesem blokowania odbywa się przez nie używany bit 7 portu o adresie 255 (#FF) wykorzystywanego do zmiany banków pamięci RAM 256 KB.

### Zmiany sprzętowe

Zmiany sprzętowe układu dysku pamięciowego ilustruje rys. 6. Jako elementu pamiętającego stan dostępności pamięci użyto przerzutnika D (F11A). Jego wejście zegarowe połączone zostało z wyjściem dekodera portu #FF (BANKEN, patrz rys. 3 — I cz. artykułu), co powoduje, że jest on widziany przez system jako 7 bit tego portu. Odpowiedni stan po włączeniu (pamięć dostępna do zapisu) wymusza układ autostartu (kondensator C1 i rezystor R2 — rys. 3 — I cz. artykułu). Bramki B33A i B34A w zależności od stanu przerzutnika F11A przepuszczają lub nie sygnał z szyny sterującej /WR mikroprocesora Z80 na wejście sterujące zapisem /WE pamięci RAM 2114 (US1, US2 — rys. 4 — I cz. artykułu) sterując dostępnością pamięci do zapisu.

UWAGA! Istniejące w wersji pierwotnej układu połączenie galwaniczne pomiędzy /WR i /WE NALEŻY ROZŁĄCZYĆ!

### Zmiany programowe

Ponieważ program obsługi dysku pamięciowego (listing zamieszczony został w II cz. artykułu) zawiera w swojej strukturze obszary buforów modyfikowane w trakcie pracy, niezbędne jest otwieranie pamięci do zapisu przed każdą akcją programu i zamykania jej po zakończeniu akcji. Sprowadza się to do następujących zmian w programie:

**1. Procedura INPUT$ (linia 90)**

```
INPUT$  LD   A, (23610)
        CP   #0B
        JR   Z,NONSNS
        CP   #17
        JR   Z,NONSNS
ERROR   LD   A,128
             ; zamknięcie pamięci do odczytu
        OUT  (#FF),A
             ; i ustawienie banku zerowego
        JR   NZ,RUNERR
             ; i dalej bez zmian
```

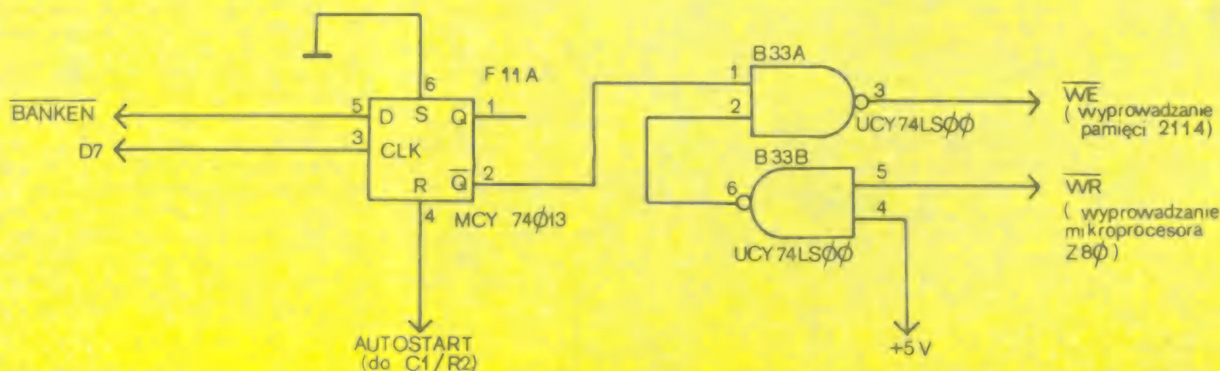
**2. Procedura RETED (linia 270)**

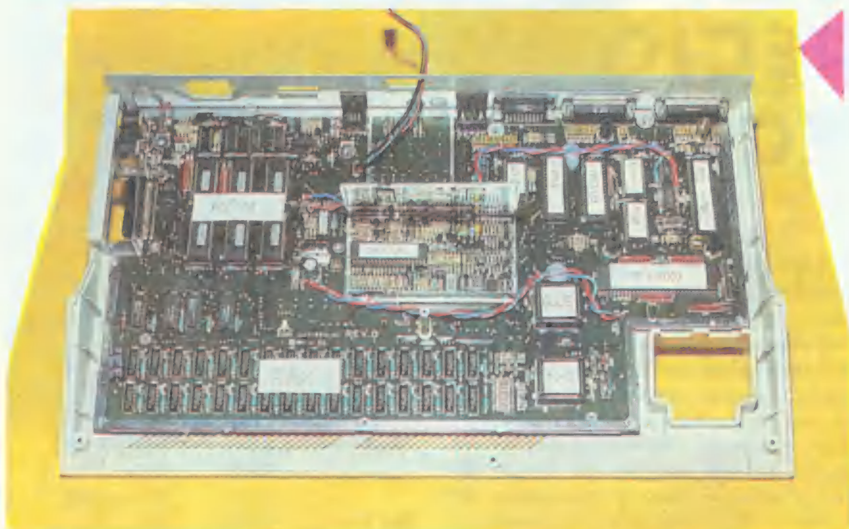
```
RETED   LD   HL,INPUT$
        PUSH HL
        LD   HL,#12B7
RETED1  PUSH HL
        LD   A, 128
             ; patrz opis proc. INPUT$
        OUT  (#FF),A
        JP   #1B76
```

**3. Procedura NONSNS (linia 350)**

```
NONSNS  XOR  A
             ; otwarcie pamięci do odczytu
        OUT  (#FF),A
             ; i ustawienie banku zerowego
        RST  18
        CP   "*"
        ...
             ; i dalej bez zmian
```

Wprowadzone zmiany spowodowały przesunięcie się miejsca startu programu; aby był on w dalszym ciągu równy 15 000 należy przenieść procedurę PRPBUF (Linie 810—860) pomiędzy procedury GETHDR (start od linii 2920) a LOAD (start od linii 3000), czyli w miejsce linii 2990. Ze względu na długość tak utworzonego kodu można usunąć z programu linie 5020 i 5030 — zawarta w nich informacja o prawach autorskich i tak znajdzie się poza kodem wynikowym obejmującym obszar 1024 B.





Wnętrze komputera ATARI 520 ST

# PRZEZ ATARI ST DO SUPERKOMPUTERA

Wprowadzenie do użytku mikrokomputerów „odebrało chleb" dużym centrom obliczeniowym wielu wyższych uczelni w zakresie stosunkowo prostych zagadnień obliczeniowych. Zjawisko to dało się zaobserwować nie tylko w krajach będących komputerowymi potęgami świata zachodniego, ale również np. w Polsce. Zupełnie nieoczekiwanie okazało się, że duża ilość takich właśnie prostych programów zajmowała w największym stopniu moc obliczeniową dużych maszyn.

Nie wszystkie jednak problemy mogą być rozwiązane przez mikrokomputer. Wszędzie tam, gdzie wykonywane są operacje na dużych zbiorach danych, złożone obliczenia macierzowe, czy też konieczne jest wykorzystanie informacji zawartych w elektronicznych bankach informacji mikrokomputer musi skorzystać z pomocy „starszego brata" — dużego komputera zainstalowanego w centrum obliczeniowym. Mikrokomputer przeistacza się wtedy w „inteligentny" terminal połączony z centrum obliczeniowym za pośrednictwem sieci komputerowej. Warto tu zwrócić uwagę na fakt, że dzięki masowej produkcji mikrokomputer osobisty jest często wielokrotnie tańszy od profesjonalnego terminala nie ustępując mu jednocześnie możliwościami. Dobrym przykładem mogą tu być komputery ATARI ST, które w dużych ilościach zostały zastosowane jako terminale w Centrum Obliczeniowym Uniwersytetu w Stuttgarcie, w RFN.

W centrum tym pracuje kilka różnych komputerów tworzących wspólnie sieć. Są tu: IBM 3081 pracujący pod kontrolą systemu operacyjnego VM/CMS, Control Data Cyber 170/835 pracujący pod systemem VMS i CONVEX 01 pracujący pod systemem UNIX. Wszystkie wymienione maszyny mogą łączyć się za pośrednictwem lokalnej sieci komputerowej Ethernet (której połączenia wykonane są na kablach światłowodowych) pracującej w systemie TCP/IP lub DECNET. Oprócz tego mogą się one łączyć z superkomputerem Cray-2 poprzez sieć HYPERchannel pracującą w systemie UNICOS.

Poprzez komputery pośredniczące (tzn. IBM 3081, Cyber 170/835, VAX 8300, VAX 780 i CONVEX C1) dostęp do superkomputera Cray-2 można uzyskać dzięki wykorzystaniu połączonych z nimi bezpośrednio ponad 600 terminali. Poza tym dzięki lokalnej sieci komputerowej Ethernet połączonej z wymienionymi wyżej komputerami pośredniczącymi z Craya-2 może korzystać ponad 100 użytkowników komputerów VAX. Przez połączenia z większymi sieciami możliwa jest wymiana danych z innymi użytkownikami uniwersyteckimi i przemysłowymi na terenie Stuttgartu.

Jak już wspomniano o decyzji zakupu ATARI ST przez Centrum Obliczeniowe Uniwersytetu w Stuttgarcie zadecydowały przede wszystkim względy ekonomiczne. Do użytku weszło tam ponad 500 egzemplarzy ATARI ST o pojemności pamięci RAM co najmniej 512 KB pamięci każdy oraz ATARI SM 124 z monitorami monochromatycznymi. Czynnikiem, który znacznie ułatwił zastosowanie ATARI ST w roli terminali było opracowanie odpowiedniego programu emulacyjnego. Program ten, którego twórcą jest młody student z Zurychu, pozwala na emulowanie funkcji terminali VT 200, VT 120 i Tektronix 4010 z nielicznymi ograniczeniami i licznymi ulepszeniami. Program ten jest ogólnie dostępny i rozpowszechniony nieodpłatnie (tzw. *public domain software*).

Podstawowym ograniczeniem ATARI ST w stosunku do typowych terminali alfanumerycznych jest to, że na ekranie może być wyświetlone tylko 128 znaków graficznych w wierszu. Poza tym jednak przed użytkownikiem korzystającym z oprogramowania UNITERM otwiera się sporo nowych możliwości:

— dla wyjść VT i Tektronix przeznaczone są osobne obszary pamięci;

— istnieje dodatkowy obszar pamięci przeznaczony do wykorzystania przy lokalnym powiększeniu fragmentu obrazu i jego demonstracji na ekranie (tzw. *zooming*). Rozdzielczość ekranu ATARI ST (640×400 punktów) jest niewiele gorsza od rozdzielczości terminala firmy Tektronix. Dzięki lokalnemu zastosowaniu powiększeń fragmentów obrazu można niemal całkowicie wyeliminować tę niedogodność, uzyskując jednocześnie szybki wgląd w szczegóły oglądanego obrazu;

— ostatnie dziesięć wierszy tekstu otrzymane z zewnątrz utrzymywane jest w specjalnym buforze, z którego mogą być dowolnie przesłane, lub odesłane z powrotem do „dużej" maszyny;

— przyciskom funkcyjnym można przypisać do dwudziestu komend. Przyspiesza to tempo pracy z dyspozycjami wykorzystywanymi najczęściej;

— zbiory dyskowe ATARI ST mogą być przenoszone w obu kierunkach tj. do i z „dużego" komputera;

— bardzo łatwo można wykonać kopię ekranu tekstowego lub graficznego na jednej z drukarek zainstalowanych w centrum obliczeniowym.

Według informacji podanych przez Lothara Ehnisa, szefa Działu Systemów Sterowanych przy Centrum Obliczeniowym Uniwersytetu w Stuttgarcie wprowadzenie ATARI ST jako terminali ma również inne zalety. Otóż wielu studentów — użytkowników uniwersyteckiego centrum obliczeniowego postanowiło zakupić ATARI ST na użytek prywatny do rozwiązywania zadań i problemów w zaciszu domowym, magazynowania danych na dyskietkach i przekazywania potem do odpowiedniego komputera. L. Ehnis twierdzi również, że w przyszłości do jednego multipleksera dołączone będzie ponad 30 ATARI ST, tworząc w ten sposób optymalne warunki pracy dla studentów, zastępując jednocześnie kosztowne terminale.

*Oprac. na podstawie „ATARI Presse Information"* **(jnkz)**

# ASEMBLER — GENS 3

## Część 7

Zauważmy, że do tej pory rozważaliśmy asemblację, podczas której program źródłowy był przechowywany w całości w pamięci operacyjnej. W przypadku asemblera GENS3 (i innych) nie jest to konieczne. Faktycznie podczas asemblacji wymagana jest obecność w pamięci operacyjnej kodu asemblera oraz zarezerwowanie wolnego obszaru na tablicę symboli i program wynikowy. Z praktyki wynika, że objętość programu źródłowego jest od 5 do 10 razy większa niż program wynikowy. Ponadto na tablicę symboli należy przeznaczyć obszar średnio 2 razy dłuższy od programu wynikowego. Zakładając, że najniższa możliwa lokacja asemblera ma adres 24 000 można wysnuć wniosek, że przechowywanie programu źródłowego w całości w pamięci operacyjnej umożliwia wygenerowanie kodu wynikowego o długości maksymalnej ok. 3 do 4 KB (patrz rys. 1). (Należy tu zaznaczyć, że 4 KB efektywnego kodu maszynowego to wielkość, w której można zamknąć rozwiązania wielu niebanalnych problemów, a panowanie nad 20 KB programu źródłowego wymaga pewnej praktyki).

Powstaje pytanie co robić, jeśli spodziewane rozmiary programu przekraczają tę granicę? W tej sytuacji należy podzielić program na segmenty i przeprowadzić asemblację programu z taśmy. Podział programu na części jest zupełnie odrębnym zagadnieniem. Można tu jedynie zauważyć, że dzielenie programu na segmenty nie powinno następować w momencie, gdy zaczyna nam brakować pamięci, lecz w fazie projektowania programu. Trudno tu podać jakieś uniwersalne rozwiązanie, bo zależy to od specyfiki danego problemu. Na ogół można jednak wyróżnić w każdym programie następujące funkcjonalnie odmienne segmenty:

1) Segment główny (sterujący);
2) Segment deklaracji zmiennych i stałych;
3) Segment procedur wejścia i wyjścia;
4) Segment procedur obliczeniowych;

mogą również wystąpić charakterystyczne dla mikrokomputerów:

5) Segment grafiki;
6) Segment muzyki.

Do asemblacji całości przystępujemy po przetestowaniu każdego segmentu osobno. Wykorzystanie asemblacji bezpośrednio z taśmy nie musi być podyktowane wyłącznie rozmiarami programu. Nawet w przypadku mniejszych programów sensowne jest opracowywanie biblioteki własnych procedur, które odpowiednio udokumentowane (nazwa, opis parametrów, działanie) przechowujemy na odrębnej taśmie. Procedury te wykorzystujemy w programie dołączając je do zasadniczego programu zapisanego w postaci źródłowej zleceniem **P**.

### Użycie zlecenia T

Dla przeprowadzenia asemblacji z taśmy niezbędne jest zapisanie na taśmie odpowiednich segmentów programu (w szczególności pojedyncza procedura może tworzyć segment) przy pomocy zlecenia **T**. Postać tego zlecenia jest następująca:

T m, n, s

gdzie:

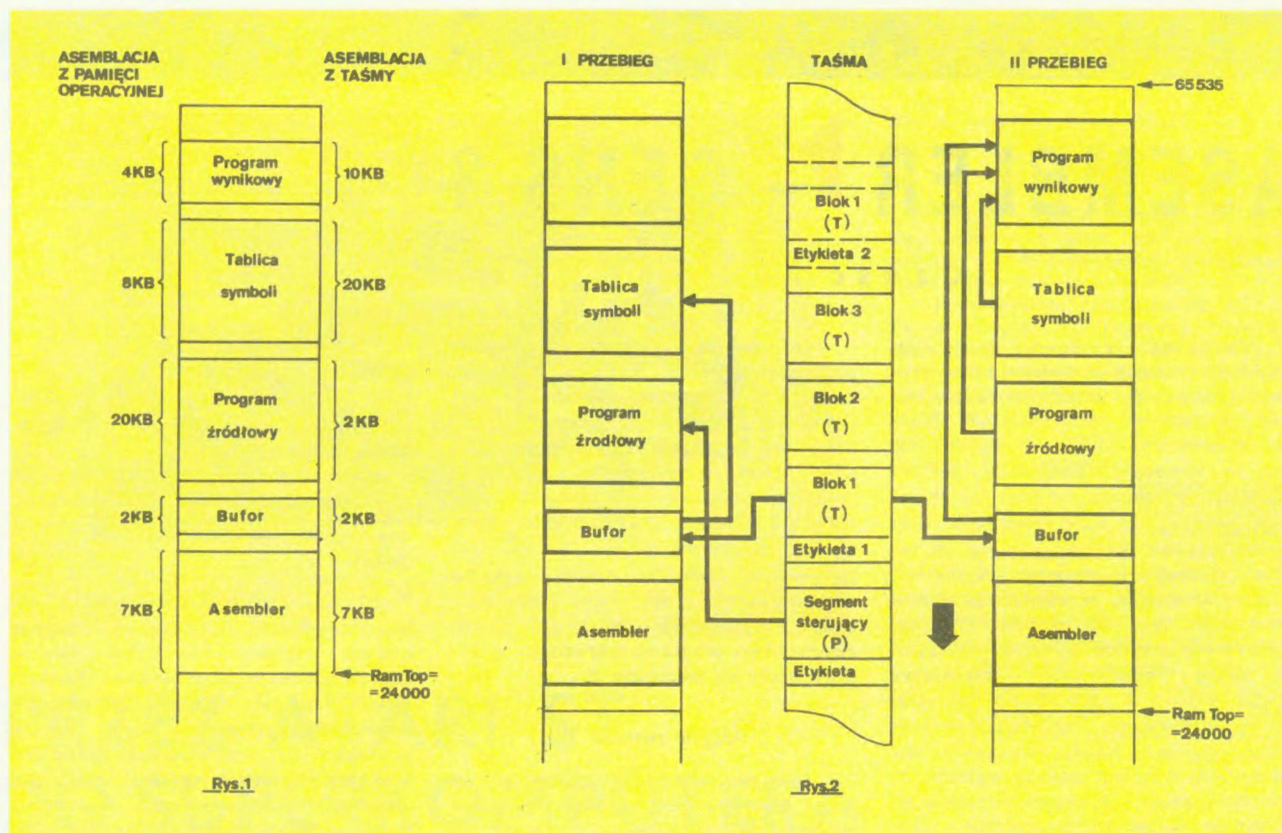
**m, n** — oznaczają numer pierwszej i ostatniej linii programu przeznaczonego do nagrania,

s — oznacza nazwę utworzonego pliku na taśmie (maksymalnie 10 znaków)

**Uwagi:**

Przed naciśnięciem klawisza ENTER należy uruchomić magnetofon. Program zostanie zapisany na taśmie blokami danych oddzielonymi przerwami. Rozmiar bloków jest stały i zależny od rozmiaru bufora dla asemblacji z taśmy zadeklarowanego po uruchomieniu asemblera. Należy więc uważać, aby przy późniejszej asemblacji tak nagranych segmentów programu źródłowego zadeklarować identyczny rozmiar bufora. Postać pliku utworzonego przy pomocy zlecenia **T** nie jest identyczna z postacią pliku utworzonego przy użyciu zlecenia **P**.






**Użycie dyrektywy F**

Do dołączenia do programu źródłowego segmentów nagranych zleceniem **T**, należy użyć w linii programowej dyrektywy **F** następującej postaci:

**Fs**

gdzie:

s — oznacza nazwę pliku zawierającego dany segment programu.

**Uwagi:**

Pominięcie nazwy spowoduje wczytanie pierwszego pliku tego typu na taśmie. Jeśli w trakcie asemblacji zostanie odczytana linia programu zawierająca dyrektywę **F** wyprowadzany jest komunikat „Start tape". Jeśli w dyrektywie **F** użyto nazwy to możliwe są dwie sytuacje:

1) Nazwa podana w dyrektywie jest zgodna z etykietą pliku odczytaną aktualnie z taśmy. Zostanie wyprowadzony komunikat „Using ..." i plik zostanie kolejnymi blokami wczytywany do bufora;

2) Aktualnie odczytana etykieta jest niezgodna z podaną w dyrektywie. Zostanie wyprowadzony komunikat „Found ..." i, niestety, plik o niezgodnej nazwie nie zostanie pominięty (jak by tego należało oczekiwać), lecz wyprowadzony zostanie komunikat „Tape error!" i asemblacja zostanie przerwana. To drobne niedopatrzenie autorów asemblera zmusza nas do nagrania na taśmie wyłącznie plików faktycznie używanych w danym przebiegu asemblacji lub ustawiania taśmy z biblioteką procedur przed początkiem każdego oczekiwanego pliku.

Również w przypadku sprzętowego błędu odczytu z taśmy nastąpi wyprowadzenie komunikatu „Tape error!" i przerwanie asemblacji (asembler wraca do edytora). Natomiast użycie w czasie odczytu klawisza BREAK powoduje powrót do BASIC-a.

**Przebieg asemblacji z taśmy**

Asemblacja następuje w dwóch przebiegach i w obydwu musi być użyta taśma z wczytywanymi segmentami. W trakcie pierwszego przebiegu analizowana jest składnia i obliczany licznik lokacji niezbędny dla wyznaczenia adresów etykiet zapamiętywanych w tablicy symboli dla każdego analizowanego aktualnie bloku danych. Proces ten odbywa się w trakcie przerw międzyblokowych. O ile pierwszy przebieg asemblacji jest poprawny asembler przechodzi do drugiego. Użytkownik musi przewinąć taśmę do początku pierwszego segmentu zapisanego zleceniem **T**. W tej fazie jest generowany program wynikowy. Różnice między przebiegiem pierwszym i drugim ilustruje rys. 2.

**Uwagi:**

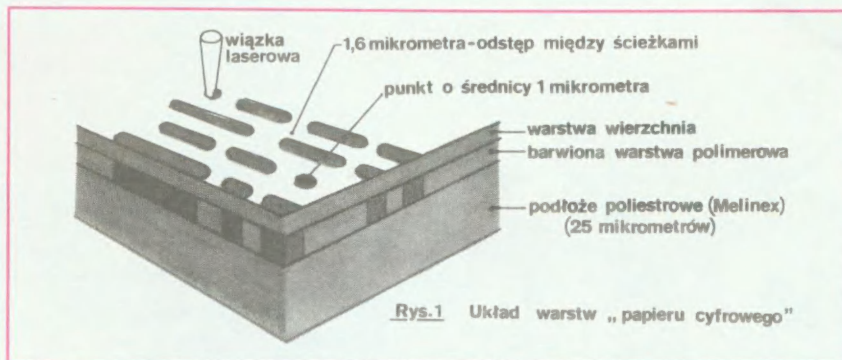
Jeśli zapisany został segment programu krótszy od zadeklarowanego rozmiaru bufora, to dla równej długości bloku zostanie on uzupełniony na taśmie zerami. Jeśli segment przeznaczony do nagrania jest dłuższy od zadeklarowanego rozmiaru bufora zostanie on podzielony na kilka bloków danych i otrzyma jedną etykietę. Ponieważ przyjęcie najmniejszego rozmiaru bufora dla asemblacji z taśmy pozwoli nam zaoszczędzić najwyżej 2KB pamięci operacyjnej wydaje się sensowne stosowanie dla własnej wygody i porządku rozmiaru maksymalnego, tym bardziej że przyspiesza to asemblację.

Oszacujmy rozmiar najdłuższego programu wynikowego możliwego do uzyskania metodą asemblacji z taśmy. Załóżmy, że segment sterujący (nagrany zleceniem **P**) składać się będzie wyłącznie z kilku linii zawierających dyrektywy **F** wczytania segmentów programu i przyjmijmy podobne jak na wstępie oszacowania dla długości obszaru tablicy symboli. Jak to ilustruje rys. 1 jest realne wygenerowanie tą metodą programu wynikowego o długości ok. 10 KB. Tym z uczestników naszego seminarium, których by ta wielkość nie satysfakcjonowała przypominamy, że cały interpreter BASIC-a ZX Spectrum zajmuje mniej niż 16 KB, a dla porównania faktyczna długość bardzo wysoko notowanego programu szachowego „Super CHESS 3.5" nie przekracza 24 KB.

Obszerny przykład rozplanowania segmentów i programu głównego do asemblacji z taśmy zostanie przedstawiony w następnym odcinku wykładu kończącym pierwszy cykl wykładów poświęcony programowaniu procesora Z80, bazujący na asemblerze GENS dla mikrokomputera ZX Spectrum.

TADEUSZ BASISTA

# „PAPIER CYFROWY”



Rys.1 Układ warstw „papieru cyfrowego”

Nową koncepcję pamięci cyfrowych wielkiej pojemności zaprezentowała firma ICI Electronic. Jest to praktycznie powrót w dziedzinie magazynowania informacji do dawnych taśm. Nowość polega jednak na stosowaniu techniki zapisu i odczytu laserowego na specjalnie spreparowanej taśmie z tzw. *papieru cyfrowego*. Podłoże taśmy stanowi warstwa wykonana z Melinexu (tworzywa sztucznego na bazie poliestrowej) grubości 25 mikrometrów. Na niej położona jest kilkumikrometrowa warstwa polimerowa, nieprzezroczysta dla światła widzialnego, mająca srebrzystą barwę. Całość pokryta jest jeszcze jedną, cienką warstwą przezroczystego tworzywa, stanowiącą ochronę dla warstwy zasadniczej. Zapis i odczyt następuje za pomocą lasera na zasadzie zmian własności optycznych odbijającej warstwy środkowej taśmy. Głowica laserowa porusza się po ścieżkach zmieniając własności optyczne warstwy, tworząc jednocześnie ślad zapisu o szerokości około 1 mikrometra. Odstęp pomiędzy ścieżkami wynosi 1,6 mikrometra, co daje łącznie 2,6 mikrometra na pełną ścieżkę. Na jednym milimetrze taśmy można zatem umieścić 385 ścieżek — nic więc dziwnego, że możliwe do uzyskania tą metodą pojemności pamięci są ogromne. Jako źródła światła do zapisu i odczytu służą najczęściej lasery półprzewodnikowe.

Oczywiście „papier cyfrowy” i wykonane na jego bazie nośniki pamięci służą jedynie do zapisu informacji i do jej odczytu. Niemożliwe jest kasowanie (na razie) zawartości zapisanej taśmy. Jednakże dzięki nieprawdopodobnie niskiej cenie do pewnych zastosowań (tam, gdzie nie jest konieczna zmiana zapisanej informacji) jest to nośnik bardzo atrakcyjny. Pierwsze próby produkcji taśmy pozwoliły na oszacowanie ceny nośnika zdolnego pomieścić 1 MB pamięci na sumę wynoszącą około 1 feniga. Jedna taśma długości 880 m i szerokości 35 mm może pomieścić... 1 terabajt (1 terabajt ≈1000 gigabajtów ≈1000 000 megabajtów — czyli pojemność równa około 1000 najbardziej pojemnych dysków twardych). Średni czas dostępu do informacji (każdego bajtu na taśmie) wynosi 28 sekund — wydaje się to dużym czasem, ale czy przy tej pojemności jakikolwiek z nośników pozwoli na taką szybkość?

Produkcją urządzeń do zapisu i odczytu zajmuje się firma CREO z Kanady. Przedstawiony w kwietniu prototyp urządzenia spełnił oczekiwania i potwierdził zalety tego sposobu przechowywania danych. Podobno gwarantowana jest już teraz trwałość 20 lat dla informacji zapisanych tym sposobem, co także jest czynnikiem przemawiającym za nową metodą zapisu.

Proponowany przebieg zapisu przedstawiono na rysunku. Zapis i odczyt dokonywany jest jednocześnie dla 32 bitów (to też ma swoje zalety). Wiązka laserowa z diody laserowej poprzez kolimator i lustro oraz układ ogniskowania naświetla taśmę z „papieru cyfrowego”. Przesuwana głowica, umieszczona w łożyskach powietrznych, poruszana jest w sposób magnetyczny. Szybkość zapisu może osiągnąć 2—10 megabajtów na sekundę, co także jest szybkością ogromną na obecne warunki.

Taśma optyczna nie jest jedyną możliwością zastosowania „papieru cyfrowego”. Oczywiście można wykonać pamięć kasetową o ogromnej pojemności, ale znacznie ciekawsze wydaje się być zastosowanie w elastycznych dyskach optycznych. W stanie statycznym „papier” nie przylega do głowicy. Dopiero po wprawieniu go w ruch obrotowy i przy tłoczeniu centralnie powietrza na skutek zjawiska Bernoulliego obracający się krążek stabilizuje swoją pozycję w bezpośredniej bliskości płaszczyzny odniesienia. Niewielka głowica dla jak najlepszego kontaktu z nośnikiem wgniata go poprzez poduszkę powietrzną. Rozwiązanie jest pewne, proste i bardzo tanie.

Jak już wspomnieliśmy, nie ma możliwości skasowania zapisu zrealizowanego tym sposobem. Zatem zastosowanie tego nośnika będzie celowe tam, gdzie zachodzi konieczność zbierania i rejestrowania dużej liczby danych. Koszt tego nośnika jest dwudziestokrotnie mniejszy od kosztu zapisu na taśmach lub dyskach sposobem magnetycznym.

Typowe zastosowania nowego zapisu to: zbieranie informacji i danych przekazywanych przez satelity, zdjęcia w medycynie, dane sejsmiczne, dane do dokumentacji archiwalnej. Ogromna pojemność pamięci i szybkość odczytu i zapisu może być drogą do rozwiązania problemu produkcji filmów z wykorzystaniem komputerów. Na razie pierwsze egzemplarze nowych urządzeń będą zastosowane w Kanadzie, a zamówione zostały przez Ministerstwo Obrony oraz Centrum Wczesnego Ostrzegania.

**Tadeusz Rzepecki**

**„Młody Technik — InforMik” wydaje Instytut Wydawniczy „Nasza Księgarnia”**

**Rada Redakcyjna:** doc. dr Zygmunt Dąbrowski, inż. Jerzy Jasiuk, dr Zygmunt Kalisz, mgr Zbigniew Słowiński, mgr inż. Jerzy Siek, dr Zbigniew Płochocki, Piotr Postawka, mgr inż. Roland Wacławek, prof. dr hab. Andrzej K. Wróblewski (przewodniczący), mgr inż. Grzegorz Zalot.

**Zespół redakcyjny:** „*InforMik*” redaguje zespół „*Młodego Technika*”. Jerzy Klawiński (sekretarz red.), Jacek Nowicki (red.), Dariusz A. Przygoda (red.), Lidia Sadowska-Szlaga (korekta), Józef Trzionka (redaktor naczelny), Roland Wacławek (software), Grzegorz Zalot (hardware), Izabela Żur (red. tech.).

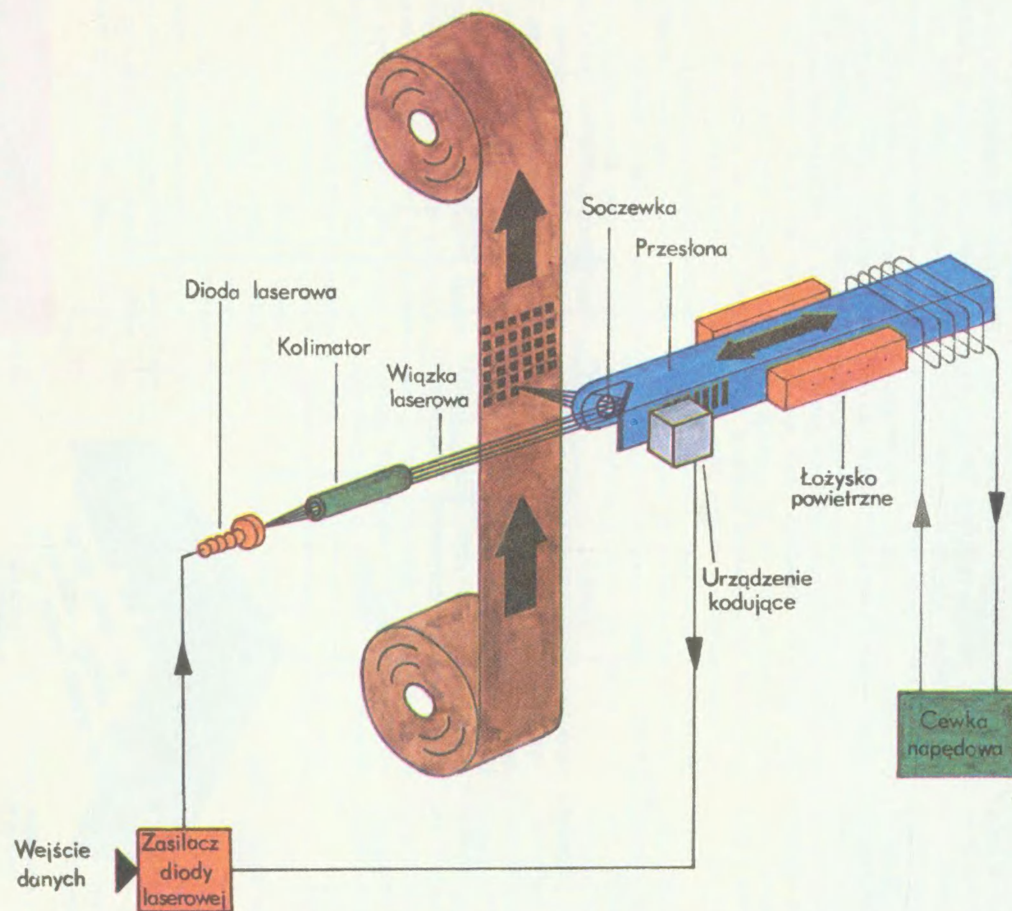
**Stali współpracownicy:** Wojciech Apel, Tadeusz Basista, Jacek Jędrzejowski, Piotr Postawka, Marek Szczepański, Krzysztof Wiśniewski.

**Adres redakcji:** ul. Spasowskiego 4, 00-389 Warszawa, lub skr. poczt. 380, 00-950 Warszawa. **Telefony:** centrala: 26-24-31 do 36. Dział Łączności z Czytelnikami — wewn. 60, pozostałe działy: wewn. 42 i 47. Redaktor naczelny: 26-26-27 lub wewn. 87.

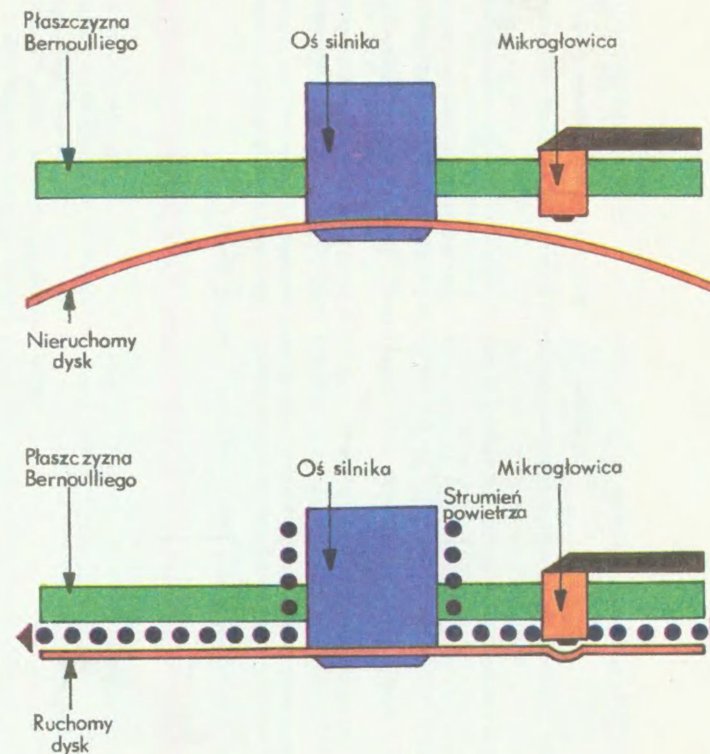
**Warunki prenumeraty:** ogólnie obowiązujące w kraju. **W STAŁEJ SPRZEDAŻY INFORMIK JEST W SALONIE WYDAWNICZYM „NASZEJ KSIĘGARNI”** ul. Spasowskiego 4 A.

Redakcja zastrzega sobie prawo adiustacji i skracania nadesłanych materiałów. Artykułów nie zamówionych redakcja nie zwraca.

Druk: Zakłady Graficzne w Katowicach. Zam. 1160/4333/8 U-6
Nakład 100 315 egz.




Rys.2 Zapis optyczny na taśmie „papieru cyfrowego”




Rys.3 Nowa propozycja budowy elastycznego dysku optycznego

Cena 120 zł Indeks nr 366013 PLISSN 0860-5696